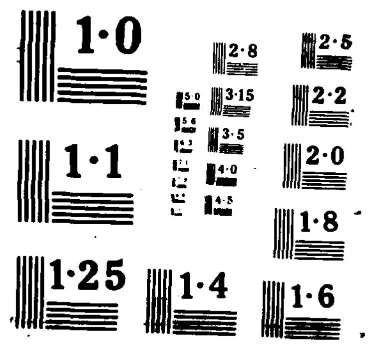
AD-A176 609 NON-SINGULAR MODELING OF RIGID MANIPULATORS(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA K MOHAMMED DEC 86 1/1

UNCLASSIFIED F/G 13/9 NL


1·0
2·8
2·5
5·0
3·15
2·2
5·6
3·5
1·1
2·0
4·0
4·5
1·8
1·25
1·4
1·6

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

NON-SINGULAR MODELING OF RIGID MANIPULATORS

by

Khayyam Mohammed

December 1986

Thesis Advisor D. L. Smith

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

AD-A176 609

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION: Naval Postgraduate School; 6b OFFICE SYMBOL (If applicable): 69	7a NAME OF MONITORING ORGANIZATION: Naval Postgraduate School
6c ADDRESS (City, State, and ZIP Code): Monterey, California 93943-5100	7b ADDRESS (City, State, and ZIP Code): Monterey, California 93943-5100
8a NAME OF FUNDING/SPONSORING ORGANIZATION; 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO; PROJECT NO; TASK NO; WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)

NON-SINGULAR MODELLING OF RIGID MANIPULATORS

12 PERSONAL AUTHOR(S)

Mohammed, Khayyam

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Masters Thesis	FROM ______ TO ______	1986 December	80

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Robotics, Robot, Simulation, Singularity, Modeling Dynamic equations of Motion, Free body analysis

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

A problem arises when conventional kinematic equations that minimize computational time are used to model a rigid revolute robot arm. Mathematical singularities result when successive link axes "line up" such that their angles are 0 or 180 degrees. This may result in erratic and uncontrollable motion of the arm until it moves away from the point of singularity. One solution is to spend a minimum amount of time at the singular position or to avoid this position altogether. Another solution is to use other sets of equations, instead of the regular resolved-rate equations, to model the robot arm. This thesis shows how using equations based on Newton's Second Principle of dynamics for a three link, two degree of freedom manipulator, the problem of singularity is avoided. The equations are demonstrated in a simulation program.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT: ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL: David Smith	22b TELEPHONE (Include Area Code): [illegible]; 22c OFFICE SYMBOL: 69Sm

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Approved for public release; distribution is unlimited.

Non-Singular Modeling of Rigid Manipulators

by

Khayyam Mohammed
Lieutenant, United States Navy
B.A. in Psychology , Queens College, 1977

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1986

Author: ______________________
Khayyam Mohammed

Approved by: ______________________
D.L. Smith, Thesis Advisor

____
A.J. Healey, Chairman,
Department of Mechanical Engineering

____
John N. Dyer,
Dean of Science and Engineering

ABSTRACT

A problem arises when conventional kinematic equations that minimize computational time are used to model a rigid revolute robot arm. Mathematical singularities result when successive link axes "line up" such that their angles are 0 or 180 degrees. This may result in erratic and uncontrollable motion of the arm until it moves away from the point of singularity. One solution is to spend a minimum amount of time at the singular position or to avoid it altogether. Another solution is to use other sets of equations, instead of the regular resolved-rate equations, to model the robot arm. This thesis shows how using equations based on Newton's Second Principle of dynamics for a three link, two degree of freedom manipulator, the problem of singularity is avoided. The equations are demonstrated in a simulation program.

TABLE OF CONTENTS

LIST OF FIGURES

TABLE OF SYMBOLS AND ABBREVIATIONS

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
A	A	Sine wave input torque data amplitude
AA	$\underset{\sim}{Aa}$	Acceleration of point a
AB	$\underset{\sim}{Ab}$	Acceleration of point b
AG1	$\underset{\sim}{Ag1}$	The acceleration vector of the center of gravity for link 1
AG2	$\underset{\sim}{Ag2}$	Same as Ag1 but for link 2
AG3	$\underset{\sim}{Ag3}$	Same as Ag1 but for link 3
AOX	aox	Linear acceleration of link zero in the x direction
AOY	aoy	Linear acceleration of link zero in the y direction
AOZ	aoz	Linear acceleration of link zero in the z direction
AX1	ax1	Linear acceleration of link 1 in the x direction
AY1	ay1	Linear acceleration of link 1 in the y direction
AZ1	az1	Linear acceleration of link 1 in the z direction

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
AX2	ax2	Linear acceleration of link 2 in the x direction
AY2	ay2	Linear acceleration of link 2 in the y direction
AZ2	az2	Linear acceleration of link 2 in the z direction
AX3	ax3	Linear acceleration of link 3 in the x direction
AY3	ay3	Linear acceleration of link 3 in the y direction
AZ3	az3	Linear acceleration of link 3 in the z direction
CTHETX(3)	cθx	A 1x3 vector Cartesian value of the angle theta for link 1-3 in the x direction, results from taking the integral of angular acceleration in the x direction twice, in degrees
CTHETY(3)	cθy	Same as cθx but in the y direction
CTHETZ(3)	cθz	Same as cθz but in the z direction
DEGRA		Conversion from degrees to radians
ETHETX(3)	θx	A 1x3 vector of euler angles for link 1-3 in thex direction, in degrees
ETHETY(3)	θy	Same as θx but in the y direction
ETHETZ(3)	θz	Same as θx but in the z direction

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
EULORY(3)	thθy3	Theoretical Euler angle for link 3 in the y direction, in degrees
ERROR(3)	Error(3)	% error between thθy3 and θy for the third link in the y direction
ERRT1X	Errt1x	% error between computed and input value of torque at joint 1
ERRT2X	Errt2x	Same as Errt1x but at joint 2
FX0	Fxo	Computed force in the x direction at joint 0
FY0	Fyo	Computed force in the y direction at joint 0
FZ0	Fzo	Computed force in the z direction at joint 0
FX1	Fx1	Computed force in the x direction at joint 1
FY1	Fy1	Computed force in the y direction at joint 1
FZ1	Fz1	Computed force in the z direction at joint 1
FX2	Fx2	Computed force in the x direction at joint 2
FY2	Fy2	Computed force in the y direction at joint 2
FZ2	Fz2	Computed force in the z direction at joint 2
G	g	Gravitational constant

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
HDX(2)	HDx	The time rate of change of angular momentum of a 2 element composite body in the x direction
HDY(2)	HDY	Same as HDx but in the y direction
HDZ(2)	HDz	Same as HDx but in the z direction
I		Counter
IA		Row dimension of matrix A and matrix B used in LEQT2F subroutine
IER		Error parameter used in subroutine LEQT2F
IDGT		Accuracy test used in subroutine LEQT2F, for iterative improvement
IXX(3,2)	Ixx	A 3x2 matrix of Moment of Inertia for the two element composite body of link 1-3 about the x axis
IYY(3,2)	Iyy	Same as Ixx but about the y axis
IZZ(3,2)	Izz	Same as Ixx but about the z axis
IXZ(3,2)	Ixz	A 3x2 matrix of Products of Inertia for the two element composite body of link 1-3 about the xz coordinate axes
IXY(3,2)	Ixy	Same as Ixz but for the xy axes
IYZ(3,2)	Iyz	Same as Ixz but for the yz axes

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
IXXA(3,2)	Ixxa	Theoretical Moment of inertia for link 3 about joint 2
JX0	jxo	Location of joint 0 in the x direction
JY0	jyo	Location of joint 0 in the y direction
JZ0	jzo	Location of joint 0 in the z direction
JX1	jx1	Location of joint 1 in the x direction
JY1	jy1	Location of joint 1 in the y direction
JZ1	jz1	Location of joint 1 in the z direction
JX2	jx2	Location of joint 2 in the x direction
JY2	jy2	Location of joint 2 in the y direction
JZ2	jz2	Location of joint 2 in the z direction
L(3,2)	L(3,2)	A 3x2 matrix that is the distance from center of link to center of mass at each link end
LCOGX(3)	LCOGx	A 1x3 location of center of gravity vector for link 1-3 in the x direction
LCOGY(3)	LCOGy	Same as LCOGx but for the y direction
LCOGZ(3)	LCOGz	Same as LCOGx but for the z direction

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
M		Used in LEQT2F subroutine as number of right hand sides
MASS(3,2)	Mass(3,2)	A 3x2 matrix of mass of each element that make up the composite body for link 1-3
MASS1	M1	Total mass of link 1
MASS2	M2	Total mass of link 2
MASS3	M3	Total mass of link 3
MATA(27,27)	MatA	A 27x27 matrix consisting of coefficients of the unknown variables
MATB(27)	MatB	A 1x27 vector consisting of the coefficient of known variables on input to subroutine LEQT2F and an output the solution to the linear equations
MI		Results from subroutine CPROD, i component of vector cross product
MJ		j component of vector cross product
MK		k component of vector cross product
MIAO, MJAO and MKAO		Cross product between wd1 and RB/G1 at joint 0, link 1, in the x, y, z direction
MIBO, MJBO and MKBO		Cross product between w1 and RB/G1 at joint 0, link 1, in the x, y, z direction

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
MIC0, MJC0 and MKC0		Cross product between w1 and MIB0, MJB0 and MKB0 at joint 0, link 1, in the x, y, z direction
MIA1, MJA1 and MKA1		Cross product between wd1 and RA/G1 at joint 1, link 1, in the x, y, z direction
MIB1, MJB1 and MKB2		Cross product between w1 and RA/G1 at joint 1, link 1, in the x, y, z direction
MIC1, MJC1 and MIC1		Cross product between w1 and MIB1, MJB1 and MKB1 respectively at joint 1, link 1, in the x, y, z direction
MIA2, MJA2 and MKA2		Cross product between wd2 and RB/G2 at joint 1, link 2, in the x, y, z direction
MIB2, MJB2 and MKB2		Cross product between w2 and RB/G2 at joint 1, link 2, in the x, y, z direction
MIC2, MJC2 and MKC2		Cross product between w2 and MIB2, MJB2 and MKB2 respectively at joint 1, link 2, in the x, y, z direction
MIA3, MJA3 and MJA3		Cross product between wd2 and RA/G2 at joint 2, link 2, in the x, y, z direction
MIB3, MJB3 and MKB3		Cross product between w2 and RA/G2 at joint 2, link 2, in the x, y, and z direction

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
MIC3, MJC3 and MKC3		Cross product between w2 and MIB3, MJB3 and MKB3 respectively at joint 2, link 2, in the x, y, z direction
MIA4, MJA4 and MKA4		Cross product between wd3 and RA/G3 at joint 2, link 3, in the x, y, z direction
MIB4, MJB4 and MKB4		Cross product between w3 and RA/G3 at joint 2, link 3, in the x, y, z direction
MIC4, MJC4 and MKC4		MJC4, MJC4 and Cross product between w3 and MIB4, MJB4, and MKB4 respectively at joint 2, link 3, at the x, y, z direction
N		Used in LEQT2F subroutine for the order of MatA and number of rows of vector B
P		Phase angle of sine wave input to joints
RUN		Number of the run conducting
RX(3,2)	Rx(3,2)	A 3x2 matrix consisting of the distance from the center of gravity of the link to center of mass for the elements of link 1-3 in the x direction
RY(3,2)	Ry(3,2)	Same as Rx(3,2) but in the y direction
RZ(3,2)	Rz(3,2)	Same as Rx(3,2) but in the z direction

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
RAG1(3)	$\underset{\sim}{r}_{a/G1}$	A 1x3 vector, distance of point a to center of gravity for link 1, in the x, y, z direction
RBG1(3)	$\underset{\sim}{r}_{b/G1}$	A 1x3 vector, distance of point b to center of gravity for link 1, in the x, y, z direction
RAG2(3)	$\underset{\sim}{r}_{a/G2}$	A 1x3 vector, distance of point a to CoG for link 2, in the x,y,z direction
RBG2(3)	$\underset{\sim}{r}_{b/G2}$	A 1x3 vector, distance of point b to CoG for link 2, in the x, y, z direction
RBG3(3)	$\underset{\sim}{r}_{b/G3}$	A 1x3 vector, distance of point b to CoG for link 3, in the x, y, z direction
SUMHDX(3)	ΣHDx	A 1x3 vector, sum of HDX for the two elements of link 1-3 in the x direction
SUMHDY(3)	ΣHDy	Same as ΣHDx but in the y direction
SUMHDZ(3)	ΣHDz	Same as ΣHDx but in the z direction
THETXR(3)		A 1x3 vector of euler angles in the x direction in radians for link 1-3
THETYR(3)		Same as THETXR(3) but in the y direction
THETZR(3)		Same as THETXR(3) but in the z direction
TOX, TOY TOZ	Tox, Toy, Toz	Input torque at joint 0 at the x, y, z direction

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
T1X, T1Y T1Z	T1x, T1y T1z	Input torque at joint 1 at the x, y, z direction
T2X, T2Y T2Z	T2x, T2y T2z	Input torque at joint 2 at the x, y, z direction
THDDOT(3)		Theoretical value of wdx for link 3 in degrees
THEORY(3)		Theoretical value of wdx for link 3 in radians
THEXR1, THEXR2, THEXR3		Second integral of wdx for links 1-3 in radians
THEYR1, THEYR2, THEYR3		Second integral of wdy for links 1-3 in radians
THEZR1, THEZR2, THEZR3		Second integral of wdz for link 1-3 in radians
TORY1X	Tory1x	Computed value of torque for joint 1
TORY2X	Tory2x	Computed value of torque for joint 2
TX1, TX2, TX3		Euler angle theta converted to radians for links 1-3 respectively in the x direction
TY1, TY2, TY3		Euler angles theta converted to radians for links 1-3 respectively in the y direction
TZ1, TZ2, TZ3		Euler angles theta converted to radians for links 1-3 respectively in the z direction
VECTA0(3) and VECTB0(3)		1x3 vector used in subroutine CPROD for joint 0

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
VECTA1(3) and VECTB1(3)		1x3 vector used in subroutine CPROD for joint 1
VECTA2(3) and VECTB2(3)		1x3 vector used in subroutine CPROD for joint 2
VECTA(3) and VECTB(3)		1x3 vector used in subroutine CPROD
W	w	Frequency of sine function input
W1, W2, and W3	W1,W2,W3	Weights of link 1, 2, and 3
WX(3)	wx(3)	A 1x3 vector of angular velocity of link 1-3 in the x direction
WY(3)	wy(3)	Same as wx(3) but in the y direction
WZ(3)	wz(3)	Same as wx(3) but in the z direction
WDX(3)	wdx(3)	Angular acceleration of link 1-3 in the x direction
WDY(3)	wdy(3)	Angular acceleration of link 1-3 in the y direction
WDZ(3)	wdz(3)	Angular acceleration of link 1-3 in the z direction
WKAREA		Work area for LEQT2F subroutine
X1, X2 and X3		Location of center of gravity for link 1-3 in the x direction
Y	y	Theoretical value of y distance from Fz2 to center of gravity of link 3

COMPUTER SYMBOL	TEXT VARIABLE	DESCRIPTION
Y1, Y2 and Y3		Location of center of gravity for link 1-3 in the y direction
Z	z	Theoretical value of z distance from Fy2 to center of gravity of link 3
Z1, Z2 and Z3		Location of center of gravity for link 1-3 in the z direction

ACKNOWLEDGEMENTS

I wish to thank the professors I had while at the Naval Postgraduate School (NPS) who taught me the basic knowledge I needed that culminated in my thesis. In particular, I would like to thank Professor Smith whose patience and help in developing the theory and coding for the simulation program lead to a successful completion of the thesis. Most of all I would like to thank my loving wife, Monique, whose encouragement and understanding helped my stint at NPS be a better one.

I. INTRODUCTION

Manipulator models which use local coordinates as a basis for simulation and control have a mathematical singularity built into them [Ref. 1]. This singularity occurs when rigid robot links align such that their relative position is either 0° or 180°. When this happens, the inverse of the Jacobian matrix becomes impossible to compute and the forward dynamics solution cannot be found.

In the control of serial link manipulators there have been various approaches which use local coordinates to achieve computational efficiency. One method deals with the Newton-Euler approach [Refs. 2, 3], another uses the Lagrangian approach [Refs. 4, 5] or there is the method of virtual work [Ref. 6]. Still another that has tried to make the solution to the dynamic equations computationally efficient by using Kane's Dynamical equations [Ref. 7]. However, although these methods have been computationally efficient, they have not been able to handle the problem of singularity [Ref. 1].

Various methods have been proposed to deal with the problem of singularity. One such method is to minimize the time near the singularity [Ref. 8], thereby reducing its effects. Another solution is to avoid the position of the manipulator that caused the singularity [Refs. 9, 10]. However, when using resolved rate equations the arm may pass through a point of singularity anyway, in response to a command [Ref. 11]. Nakamura and Hanafusa [Ref. 12] have proposed to determine the joint motion for the requested motion of the end effector by evaluating the feasibility of the joint motion. This determined joint motion is called an inverse kinematic solution with singularity robustness. Other solutions deal with presenting equations that can translate the manipulators in the neighborhood of singularity [Refs.13,14] and in identifying geometric singular positions [Ref. 15].

It has also been shown that redundancy of robot manipulators is effective in dealing with singularities [Ref. 16]. Klein and Huang [Ref. 17] have studied the method of pseudo-inverse control, for use with redundant manipulators, with recommendations for improvement. Uchiyama [Ref. 18] proposed switching the control mode in the neighborhood of singular points from the mode using inverse kinematics to the joint control mode. A seven degree of freedom kinematic design with a spherical shoulder joint was proposed [Refs. 19, 20], as well as a seven joint robot [Ref. 21] to handle singularity. A four degree of freedom wrist was studied to overcome wrist singularity [Ref. 22]. Shih [Ref. 23] looked at the physical quantities and combinations of physical quantities which are unaffected by redundancy to simplify the solution of a redundant system. However, even though there are some redundant manipulators constructed [Refs. 24, 25], research cannot do away with singularities, and so consideration still has to be given to the control of the manipulator in case of inadvertant singularities.

This paper will derive equations of motion using the First Principles of Newtonian dynamics in terms of global coordinates in order to eliminate the problem of singularity. By the method of free body analysis, each link of the manipulator is treated as if it were a free body with forces and moments applied at the joints. Only revolute joints will be considered. Although tedious and time consuming (computer time), this paper will show by simulation how the problem of singularity may thus be overcome.

II. ROBOT MODELLING AND SIMULATION PROBLEM

This thesis does not deal with the control aspect of rigid, revolute linkages but rather the mathematical dynamic modelling. Given the dynamic model, the link masses and inertia properties, initial link alignments, and joint torques, then the joint forces, acceleration, velocity and position can be predicted via a simulation program. In the present approach, all dynamic properties except for acceleration and forces were assumed to remain constant over a simulated time interval. This assumption linearized the equation of motion so that a simple matrix inversion could be used to solve for the unknowns. As shown in Figure 1, the simulation is updated with the predicted velocity ($\dot{\underline{\theta}}$) and position ($\underline{\theta}$), following integration at the next time step. Simulation validation is done by comparing the theoretical position (thθy3) to the predicted position (θy3) for link 3 and actual torque (T1x, T2x) to computed torque (Tory1x, Tory2x) for links 2 and 3.

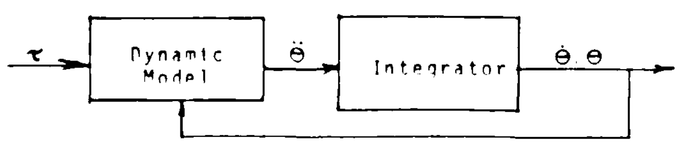


Figure 1. Manipulator Simulation Block Diagram

III. THEORETICAL DEVELOPMENT

A. MANIPULATOR ARM CONFIGURATION

This thesis develops a generalized simulation program for a robot manipulator that is a serial connection of three rigid links, jointed by one-degree of freedom revolute joints. Joint actuators are assumed to be located between successive links to apply the torque necessary to position the link.

B. THEORY

The method of solution is based on the principle of free body analysis. For this approach each body of the three link manipulator is treated as if it were a free body with forces and moments applied at each joint, as shown in Figure 2. The global cartesian coordinate system X, Y, Z as well as force and moment torque conventions are also evident in the figure. Note that a local coordinate system, that is a coordinate system that is local to each joint, will not be used but rather a single global system will be

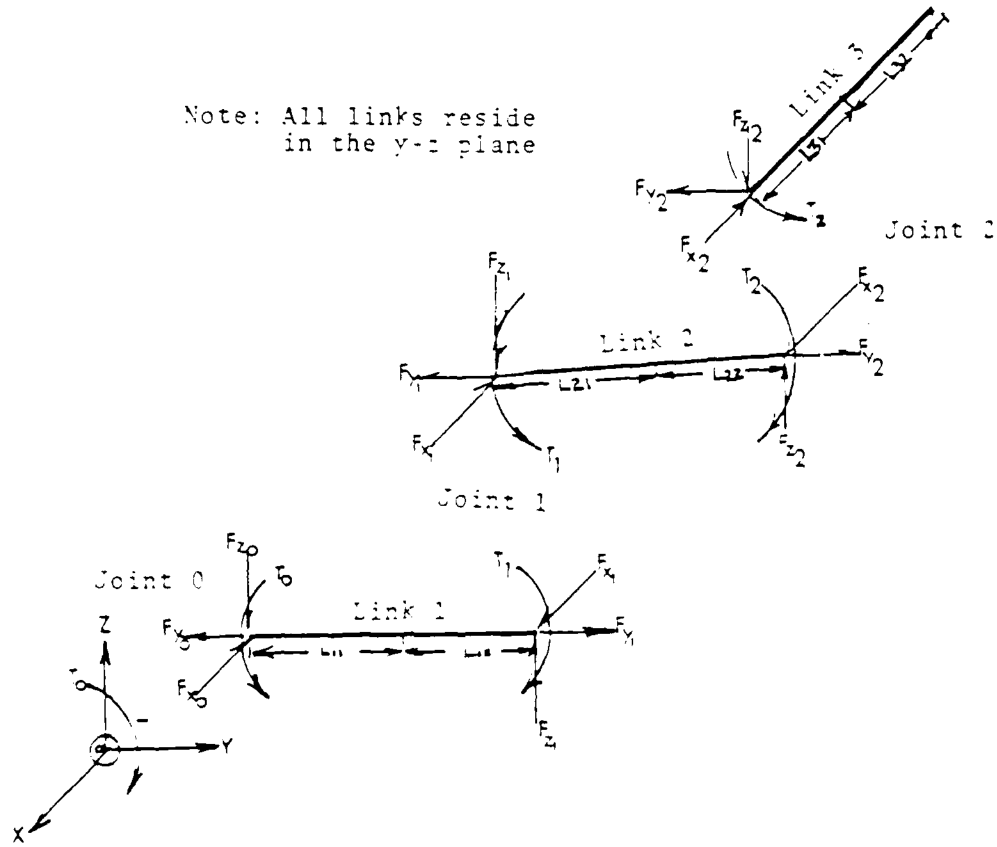


Figure 2. Free Body Diagrams

adopted. So all positions, distances, etc., will be referenced to the base of the manipulator system which will be at joint zero. The effects of flexibility of the robot manipulator will not be considered since ideal, rigid bodies are assumed.

In developing the dynamic equations of motion for each link Newton's Second Principle of Motion is used. The known variables are torque at the joints, mass of each link, linear acceleration of joint zero, initial angular acceleration, angular velocity and position of all links. The unknowns are the forces at the joints, linear and subsequent angular accelerations of the links.

C. DYNAMIC EQUATIONS OF MOTION OF LINK ONE

1. Sum of Forces Equations

In the free body analysis of link one (Figure 2) the sum of the forces in the x direction is:

$$\Sigma Fx = Fx1 - Fx0 = M1ax1 \tag{1}$$

Similarly sum of the forces in the y direction is:

$$\Sigma Fy = Fy1 - Fy0 = M1ay1 \tag{2}$$

and the sum of the forces in the z direction is:

$$\Sigma Fz = Fz1 - Fz0 - W1 = M1az1 \tag{3}$$

2. Joint Equations

We begin by evaulating the joint equations at joint zero (Ref. 26, equation (8/4), pp. 423). If the joint is sequested and analysis conducted at a point on link zero (subscript a) and another at a point on link one (subscript b) that is common to both, so when linked together they are equal. This results in two equations and the two unknowns

wd1 and w1.

As a result:

$$\underset{\sim}{Aa} = \underset{\sim}{Ao}$$

which is the acceleration at joint zero,

and

$$\underset{\sim}{Ab} = \underset{\sim}{A1} + (\underset{\sim}{wd1} \times \underset{\sim}{rb/G1}) + \underset{\sim}{w1} \times (\underset{\sim}{w1} \times \underset{\sim}{rb/G1})$$

which is the acceleration of point b on joint one. Here rb/G1 is the distance from point b to the center of gravity of link one, and A1 is the acceleration at the center of mass of link one or,

$$\underset{\sim}{rb/G1} = (jx0 - LCOGx1)i + (jy0 - LCOGy1)j + (jz0 - LCOGz1)k$$
$$= rb/G1x + rb/G1y + rb/G1z$$

After equating Aa and Ab and having the known variables on the right side of the equation and unknown variables on the left side the following sets of equations result:

$$Ax1 + wdy1(rb/G1z) - wdz1(rb/G1y) = Aox - MICO \qquad (4)$$

where MICO equals

$$=wy1wx1(rb/G1y)-w^2y1(rb/G1x)-w^2z1(rb/G1x)$$

$$+ wz1wx1(rb/G1z)$$

also

$$Ay1 + wdz1(rb/G1x)-wdx1(rb/G1z)=Aoy-MJC0 \quad (5)$$

where MJC0 equals

$$=wz1wy1(rb/G1z)-w^2z(rb/G1y)-w^2x1(rb/G1y)$$

$$+ wx1wy1(rb/G1x)$$

and

$$Az1 + wdx1(rb/G1y)-wdy1(rb/G1x)=Aoz-MKC0 \quad (6)$$

MKC0 equals

$$=wx1wz1(rb/G1x)-w^2x1(rb/G1z)-w^2y1(rb/G1z)$$

$$+ wy1wz1(rb/G1y)$$

3. Sum of Moment Equations

Computing the sum of the moment equations about the center of gravity results in:

$$\Sigma \underset{\sim}{M1}=(r\underset{\sim}{0/}G1 \times \underset{\sim}{F0}) + (r1\underset{\sim}{/}G1 \times \underset{\sim}{F1})-\underset{\sim}{T1} + \underset{\sim}{T0}$$

where the vector $r\underset{\sim}{0/}G1$ is the distance from joint zero to the center of gravity of link one and vector $r1\underset{\sim}{/}G1$ is the distance from joint one to the center of gravity of link one, in the x, y and z directions. Such that

$$\underset{\sim}{r0}/G1 = r\underset{\sim}{jo}-r\underset{\sim}{G1}$$

and

$$\underset{\sim}{r1}/G1 = r\underset{\sim}{j1}-r\underset{\sim}{G1}$$

so

$$r\underset{\sim}{j0}-r\underset{\sim}{G1}=(xj0-xG1)i + (yj0-yG1)j + (zj0-zG1)k$$

and

$$rj1 - rG1 = (xj1 - xG1)i + (yj1 - yG1)j + zj1 - zG1)k$$

In the x, y and z directions the sum of moment equations are:

$\Sigma M1$ in x direction=

$$(-yj0/G1)Fz0 + (zj0/G1)Fy0 + (yj1/G1)Fz1 - (zj1/G1)Fy1$$
$$-T1x + T0x \qquad (7a)$$

$\Sigma M1$ in y direction=

$$(-zj0/G1)Fx0 + (xj0/G1)Fz0 + (zj1/G1)Fx1 - (xj1/G1)Fz1$$
$$-T1y + T0y \qquad (8a)$$

$\Sigma M1$ in z direction=

$$(-xj0/G1)Fy0 + (yj0/G1)Fx0 + (xj1/G1)Fy1 - (yj1/G1)Fx1 - T1z$$
$$+ T0z \qquad (9a)$$

From Ref. 26, equation (517) pp.227 the sum of the moments about a fixed point that does not move with the body is equal to the time rate of change of angular momentum of the system ($\dot{H}$) about the fixed point, $\Sigma M = \dot{H}$. In the present study we have let each link be a composite body of two elements. The angular momentum (H) for a composite body where the number of elements of the body is two, about the center of gravity of each link is $Hi = \sum_{i=1}^{2} [Ri \times (w \times Ri)]Mi$, where Ri is the distance from the center of gravity of each link to the appropriate element (1or2) in the x, y and z direction. So:

$$Hx = \sum_{i=1}^{2} [Ryi(wx(Ryi) - wy(Rxi)) - Rzi(wz(Rxi)wx - (Rzi))]Mi$$

$$Hx = [R^2y1(wx) - Ry1(Rx1)(wy) - Rz(Rx1)(wz) + R^2z1(wx)]M1$$
$$+[R^2y2(wx) - Ry2(Rx2)(wy) - Rz2(Rx2)(wz) + (R^2z2)wx]M2$$

If $Ixx = \int Ry^2 + Rz^2 \, dm$,

and $Ixy = \int RxRy \, dm$,

and

$Ixz = \int RxRy \, dm$,

then:

$$Hx = [I1xx(wx) - I1xy(wy) - I1xz(wz)]M1$$

$$+ [I2xx(wx) - I2xy(wy) - I2xz(wz)]M2.$$

and

$$HDx = [I1xx(wdx) - I1xy(wdy) - I1xz(wdz)]M1$$

$$+ [I2xx(wdx) - I2xy(wdy) - I2xz(wdz)]M2 \qquad (7b)$$

by assuming the moment of inertia does not change with time but is constant for a given time interval.

By similar analysis it can be shown:

$$Hy = \sum_{i=1}^{2} [Rzi(wy(Rzi) - wz(Ryi)) - Rxi(wx(Ryi) - wy(Rxi))]Mi$$

and if $Iyy = \int Rx^2 + Rz^2 \, dm$,

and $Iyz = \int RyRz \, dm$,

and $Ixy = \int RxRy \, dm$

then:

$$HDy = [I1yy(wdy) - Iiyz(wdz) - Iiyx(wdx)]Mi$$

$$+ [I2yy(wdy) - I2yz(wdz) - I2yx(wdx)]M2 \qquad (8b)$$

and

$$Hz = \sum_{i=1}^{2} [Rxi(wz(Rxi) - wx(Rzi)) - Ryi(wy(Rzi) - wz(Ryi))]Mi.$$

If $Izz = \int Rx^2 + Ry^2 \, dm$,

So

$$Hz=[I1zz(wz)-I1yz(wy)-I1zx(wx)]M1 + [I2zz(wz)-I2yz(wy)-I2zx(wx)]M2$$

then

$$HDz=[I1zz(wdz)-I1yz(wdy)-I1zx(wdx)]M1 + [I2zz(wdz)-I2yz(wdy)-I2zx(wdx)]M2 \qquad (9b)$$

Combining equations (7a) and (7b) and keeping known variables on the right side and unknown variables on the left side yields:

$$\Sigma M1x= (-yjo/G1)Fzo + (zjo/G1)Fyo + (yj1/G1)Fz1 - (zj1/G1)Fy1-HDx=T1x-Tox \qquad (7)$$

Combining equations (8a) and (8b) yields:

$$\Sigma M1y=(-zjo/G1)Fxo + (xjo/G1)Fzo + (zj1/G1)Fx1 - (xj1/G1)Fz1-HDy=T1y-Toy \qquad (8)$$

combining equations (9a) and (9b) yields:

$$\Sigma M1z=-(xjo/G1)Fyo + (yjo/G1)Fxo + (xj1/G1)Fy1 - (yj1/G1)Fx1-HDz=T1z-Toz \qquad (9)$$

D. LINK TWO EQUATIONS

1. Sum of Forces Equations

From the free body diagram (Figure 2) it follows that

$$\Sigma Fx=Fx2 - Fx1=M2ax2 \qquad (10)$$

$$\Sigma Fy=Fy2 - Fy1=M2ay2 \qquad (11)$$

$$\Sigma Fz=Fx2 - Fz1=M2az2 \qquad (12)$$

2. Joint Equations

Analysis is conducted at joint one where similar equations are used as in joint zero with a point on link one (a) and one on link two (b). For point a the equation is

$$\tilde{Aa} = \tilde{A1} + \tilde{wd1} \times \tilde{ra/G1} + \tilde{w1} \times (\tilde{w1} \times \tilde{ra/G1})$$

$\tilde{ra/G1}$ is a vector whose distance is measured from point a to the center of gravity of link one in the x, y and z direction.

$$\tilde{ra/G1} = (jx1 - LCOGx1)i + (jy1 - LCOGy1)j + (jz1 - LCOGz1)k$$
$$= ra/G1x + ra/G1y + ra/G1z$$

For point b the equation is:

$$\tilde{Ab} = \tilde{A}\ 2 + \tilde{wd2} \times \tilde{rb/G2} + \tilde{w2} \times (\tilde{w2} \times \tilde{rb/G2})$$

where $\tilde{rb/G2}$ is a vector whose distance is measured from point b to the center of gravity of link two.

$$\tilde{rb/G2} = (jx1 - LCOGx2)i + (jy1 - LCOGy2)j + (jz1 - LCOGz2)k$$
$$= rb/G2x + rb/G2y + rb/G2z$$

Equating $\tilde{Aa}$ and $\tilde{Ab}$ and setting knowns and unknowns on the respective sides of the equation results in:

$$Ax2 - Ax1 + wdy2(rb/G2z) - wdz2(rb/G2y) - wdy1(ra/G1z) + wdz1(ra/G1y) = MIC1 - MIC2 \qquad (13)$$

$$MIC1 = wy1wx1(ra/G1y) - w2y1(ra/G1x) - w2z1(ra/G1x) + wz1wx1(ra/G1z)$$

$$MIC2 = wy2wx2(ra/G2y) - w2y2(ra/G2x) - w2z2(rb/G2x) + wz2wx2(rb/G2z)$$

$$\mathrm{Ay2-Ay1 + wdz2(rb/G2x)-wdx2(rb/G2z)-wdz1(ra/G1x)}$$
$$\mathrm{+ wdx1(ra/G1z)=MJC1-MJC2} \tag{14}$$
$$\mathrm{MJC1=wz1wy1(ra/G1z)-w2z1(ra/G1y)-w2x1(ra/G1y)}$$
$$\mathrm{+ wx1wy1(ra/G1x)}$$
$$\mathrm{MJC2=wz2wy2(rb/G2z)-w2z2(rb/G2y)-w2x2(rb/G2y)}$$
$$\mathrm{+ wx2wy2(rb/G2x)}$$
$$\mathrm{AZ2-AZ1 + wdx2(rb/G2y)-wdy2(rb/G2x)-wdx1(ra/G1y)}$$
$$\mathrm{+ wdy1(ra/G1x)= MKC1-MKC2} \tag{15}$$
$$\mathrm{MKC1=wx1wz1(ra/G1x)-w2x1(ra/G1z)-w2y1(ra/G1z)}$$
$$\mathrm{+ wy1wz1(ra/G1y)}$$
$$\mathrm{MKC2=wx2wz2(rb/G2x)-w2x2(rb/G2z)-w2y2(rb/G2z)}$$
$$\mathrm{+ wy2wz2(rb/G2y)}$$

3. <u>Sum of the Moment Equations</u>

These equations have a similar development as that of link one:

$$\Sigma \underset{\sim}{\mathrm{M2}} = (\mathrm{rj}\underset{\sim}{\mathrm{1/}}\mathrm{G2}) \times \underset{\sim}{\mathrm{F1}} + (\mathrm{rj}\underset{\sim}{\mathrm{2/}}\mathrm{G2}) \times \underset{\sim}{\mathrm{F2}} + \underset{\sim}{\mathrm{T1}} - \underset{\sim}{\mathrm{T2}}$$

where

$$\mathrm{rj}\underset{\sim}{\mathrm{1/}}\mathrm{G2=(xj1-xG2)i + (yj1-yG2)j + (zj1-zG2)k}$$
$$\mathrm{rj}\underset{\sim}{\mathrm{2/}}\mathrm{G2=(xj2-xG2)i + (yj2-yG2)j + (zj2-zG2)k}$$
$$\mathrm{\Sigma M2x= -(yj1-yG2)Fz1 + (zj1-zG2)Fy1 + (yj2-yG2)Fz2}$$
$$\mathrm{- (zj2-zG2)Fy2 + T1x-T2x} \tag{16a}$$
$$\mathrm{\Sigma M2y= -(zj1-zG2)Fx1 + (xj1-xG2)Fz1 + (zj2-zG2)Fx2}$$
$$\mathrm{- (xj2-xG2)Fz2 + T1y-T2y} \tag{17a}$$
$$\mathrm{\Sigma M2z=- (xj1-xG2)Fy1 + (yj1-yG2)Fx1 + (xj2-xG2)Fy2}$$
$$\mathrm{- (yj2-yG2)Fx2 + T1z-T2z} \tag{18a}$$

From angular momentum equation developed for link one, it can be shown for link two:

$$\Sigma M2x = HDx \quad (16b)$$

$$\Sigma M2y = HDy \quad (17b)$$

$$\Sigma M2z = HDz \quad (18b)$$

Combining equations (16a) and (16b) the following result:

$$-(yj1-yG2)Fz1 + (zj1-zG2)Fy1 + (yj2-yG2)Fz2-(zj2-zG2)Fy2$$
$$-HDx = -T1x + T2x \quad (16)$$

Combining equations (17a) and (17b) yield the following result:

$$-(zj1-zG2)Fx1 + (xj1-xG2)Fz1 + (zj2-zG2)Fx2-(xj2-xG2)Fz2$$
$$-HDy = -T1y + T2y \quad (17)$$

Combining equations (18a) and (18b) yield the following result:

$$-(xj1-xG2)Fy1 + (yj1-yG2)Fx1 + (xj2-xG2)Fy2-(yj2-yG2)Fx2$$
$$-HDz = -T1z + T2z \quad (18)$$

E. LINK THREE EQUATIONS

1. Sum of Forces Equations

Following similar logic from that previously developed:

$$\Sigma Fx = -Fx2 = M3ax3 \quad (19)$$

$$\Sigma Fy = -Fy2 = M3ay3 \quad (20)$$

$$\Sigma Fz = -Fz2 - W3 = M3az3 \quad (21)$$

2. Joint Equations

With point a on link two and point b on link three one gets for joint equations at joint two:

$$\underset{\sim}{Aa}=\underset{\sim}{A2} + (\underset{\sim}{wd2} \times \underset{\sim}{ra/G2}) + \underset{\sim}{w2} \times (\underset{\sim}{w2} \times \underset{\sim}{ra/G2})$$

where $\underset{\sim}{ra/G2}$ is a vector whose distance is measured from point a to center of gravity of link two in the x,y and z direction.

$$\underset{\sim}{ra/G2}=(jx2-LCOGx2)i + (jy2-LCOGy2)j + (jz2-LCOGz2)k$$
$$=ra/G2x + ra/G2y + ra/G2z$$

For point b

$$\underset{\sim}{Ab}=\underset{\sim}{A3} + \underset{\sim}{wd3} \times \underset{\sim}{rb/G3} + \underset{\sim}{w3} \times (\underset{\sim}{w3} \times \underset{\sim}{rb/G3})$$

where $\underset{\sim}{rb/G3}$ is a vector whose distance is measured from point b to center of gravity of link three in the x, y and z direction.

$$\underset{\sim}{rb/G3}=(jx2-LCOGx3)i + (jy2-LCOGy3)j + (jz2-LCOGz3)k$$
$$=rb/G3x + rb/G3y + rb/G3z$$

Equating $\underset{\sim}{Aa}$ and $\underset{\sim}{Ab}$ and setting knowns and unknowns on the respective sides of the equation results in:

$$Ax3-Ax2 + wdy3(rb/G3z)-wdz3(rb/G3y)-wdy2(ra/G2z) + wdz2(ra/G2y)=MIC3-MIC1 \qquad (22)$$

$$MIC3=wy2wx2(ra/G2y)-w2y2(ra/G2x)-w2z2(ra/G2x) + wz2wx2(ra/G2z)$$

$$MIC4= wy3wx3(rb/G3y)-w2y3(rb/G3x)-w2z3(rb/G3x) + wz3wx3(rb/G3z)$$

$$Ay3 -Ay2 + wdz3(rb/G3x)-wdx3(rb/G3z)-wdz2(ra/G2x) +wdx2(ra/G2z)=MJC3-MJC4 \qquad (23)$$

$$MJC3=wz2wy2(ra/G2z)-w2z2(ra/G2y)-w2x2(ra/G2y) + wx2wy2(ra/G2x)$$

MJC4=wz3wy3(rb/G3z)-w2z3(rb/G3y)-w2x3(rb/G3y)

+ wx3wy3(rb/G3x)

AZ3-AZ2 + wdx3(rb/G3y)-wdy3(rb/G3x)-wdx2(ra/G2y)

+ wdy2(ra/G2x)= MKC3-MKC4 (24)

MKC3=wx2wz2(ra/G2x)-w2x2(ra/G2z)-w2y2(ra/G2z)

+ wy2wy2(ra/G2y)

MKC4=wx3wz3(rb/G3x)-w2x3(rb/G3z)-w2y3(rb/G3z)

+ wy3wz3(rb/G3y)

3. Sum of Moment Equations

As in the development of the equations for link one:

ΣM3=(rj2/G3) X F2 + T2

where rj2/G3=(xj2-xG3)i + (yj2-yG3)j + (zj2-zG3)k

=xj2/G3 + yj2/G3 + zj2/G3

ΣM3x=(-yj2/G3)Fz2 + (zj2/G3)Fy2 + T2x (25a)

ΣM3y= (-zj2/G3)Fx2 + (xj2/G3)Fz2 + T2y (26a)

ΣM3z=(-xj2/G3)Fy2 + (yj2/G3)Fx2 + T2z (27a)

From the angular momentum theory:

ΣM3x=HDx (25b)

ΣM3y=HDy (26b)

ΣM3z=HDz (27b)

Combining equations (25a) and (25b) the following results:

(-yj2/G3)Fz2 + (zj2/G3)Fy2-HDx= -T2x (25)

Combining equations (26a) and (26b) the following results:

((-zj2/G3)Fx2 +(xj2/G3)Fz2-HDy= -T2y (26)

Combining equations (27a) and (27b) the following results:

(-xj2/G3)Fy2 + (yj2/G3)Fx2-HDz= -T2z (27)

IV. COMPUTATIONAL APPROACH

The language chosen to write the program was the Digital Simulation Language (DSL) using Fortran 77 coding. This language does an excellent dynamic simulation that allows the user to be interactive, with real time processing vice batch mode processing commonly used with the Continuous System Modelling Program (CSMP), and all calculations done in double precision. The source code produced for this program was complied on an IBM 3033 computer using a Fortran 77 compiler.

A. PRINCIPLE PROGRAM MATRICIES

A 27x27 Matrix A (MatA) was created from the coefficients of the unknowns (forces, linear acceleration and angular acceleration) from equations (1) to (27). Correspondingly a 27x1 Matrix B (MatB) was generated from equations (1) to (27) from all knowns (torques, angular velocities, link masses, and various positions). Subroutine CPROD was used to conduct all cross products required in the main program. Subroutine (LEQT2F) was then called from the IMSL library. This subroutine takes MatA inverts it, multiplies it by MatB and solves the generalized equation $A\underset{\sim}{x}=\underset{\sim}{b}$ for the $\underset{\sim}{b}$ vector using Gaussian elimination with iterative improvement to get accuracy within six decimal digits. The output from LEQT2F returns from the subroutine via MatB. This output is then used by the INTGRL DSL statement to take the integral of angular acceleration (wdx, wdy, wdz) to get angular velocity (wx, wy, wz) and again to get the position of the link with respect to theta (cθx, cθy, cθz) for each torque input per time step. The cartesian orientations are converted to Euler angles (θx, θy, θz) prior to returning to the beginning of the program. The method used to solve the second order differential, equation for accelerations is invoked by ADAMS which is the second order, variable step integration ADAMS method. This method was shown to be the fastest (CPU time) and most accurate of the methods available [Ref. 8]. Similarly, INTGRL is applied to find the linear acceleration ($\underset{\sim}{A}$) of each link, velocity ($\underset{\sim}{V}$) and finally the position of the center of gravity of the link. These newly found values are fed back into the beginning of the simulation program for the next time step until the end of the interval. This process is summarized in Figure 3.

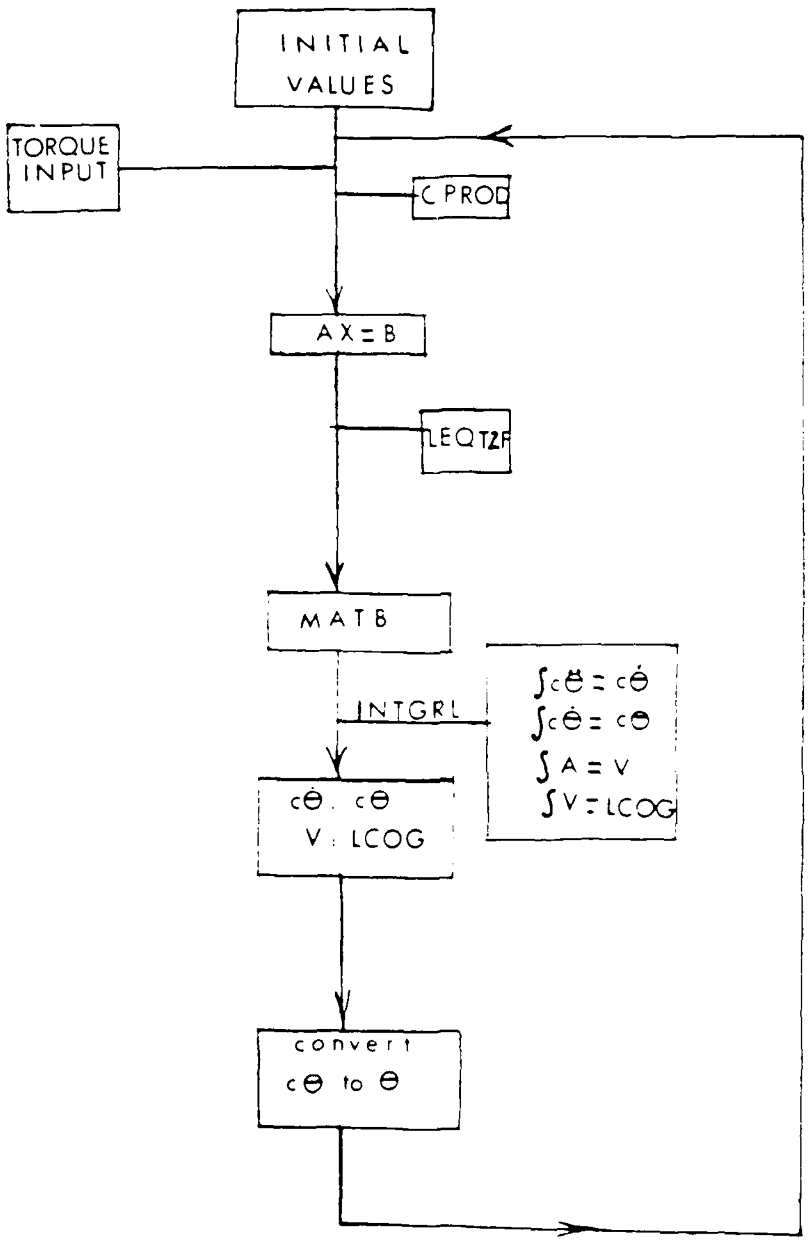


Figure 3. Computer Simulation Block Diagram

Constraints are also built into the simulation program that enable the operator to limit the movement of the links in the yz plane for a two dimensional demonstration of link-three-only movement. The constraints are also used for link two and three movement. The constraints are applied by zeroing out a row, except for the diagonal which is set to 1.0. The MatB entry is set to the constrained value.

It is during this simulation that a differentiation should be made between the cartesian theta ($c\theta$) position developed by the INTGRL function (Figure 4a) and the Euler angler theta (θ) used as direction angles in computing distances (Figure 4b). When in the yz plane the angular acceleration is about the x axis and when the integral is taken twice with respect to time, what results is the angle theta about the x axis. This cartesian angle is defined as $c\theta x$ and is obviously not the same as the theta angle used to position the link initially which is defined as θy. To resolve this discrepancy when $c\theta x$ is computed it is converted to the euler angle θy by setting the two equal so $\theta y = c\theta x$, in a two-dimensional simulation. Additionally, euler angle $\theta z = 90^\circ - \theta y$ and $\theta x = 90^\circ$ whenever simulating, two-dimensional yz plane motions.

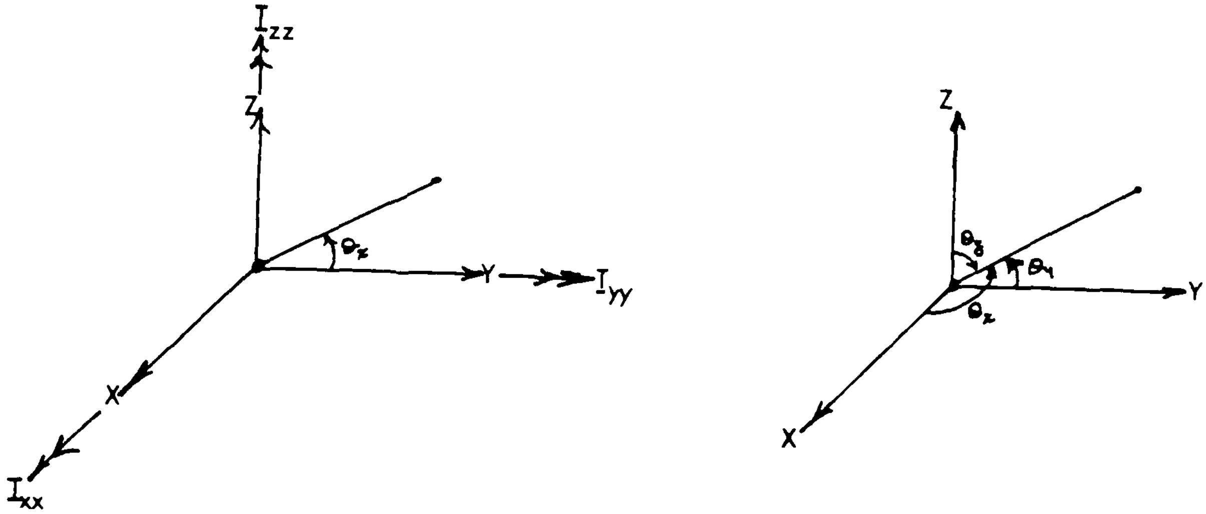


Figure 4. Angles of Orientation (a) Cartesian angles (b) Euler angles.

B. PROGRAM VALIDATION

Validation of the simulation program takes place in two ways.

1. Validation of One Link Case

For link three the theoretical value of theta in the x direction (thθx3), is compared to the value of theta in the x direction (θx3) that the simulation computed for each time step (Appendix A has the program listing).

As a test case, the torque delivered at joint two was assumed to be:

$$T2x = 10\sin(2\pi t).$$

Also, Ixxa, the moment of inertia about joint two, is equal to mass at the end of the link M(3,2) times the distance from joint two to the mass at the end of the link, squared or,

$$Ixxa = M(3,2) \times (L(3,2) + L(3,1))^2.$$

This may be used to solve for thθx by taking the integral with respect to the time which results in:

$$th\ddot{\theta}x3 = \frac{T2x}{Ixxa}$$

$$\int_0^t th\ddot{\theta}x3\,dt = \frac{-10}{Ixxa(2\pi)}\cos(2\pi t)\Big|_0^t$$

$$th\dot{\theta}x3 = \left(\frac{-10}{2\pi}(\cos 2\pi t) + \frac{10}{2\pi}\right)\frac{1}{Ixxa}$$

$$\int_0^t th\dot{\theta}x3\,dt = th\theta x3 = \left(\frac{-10}{4\pi^{**}2}\sin(2\pi t) + \frac{10t}{2\pi}\right)\frac{1}{Ixxa} + \frac{\pi}{4}$$

For comparison % error is used so

$$\% \text{ error} = \frac{(th\theta 3 - \theta x3)}{\max th\theta x3} \times 100$$

2. Validation of Two Links Case

For the validation of two links the computed torques at joints two and joint one (Tory2x, Tory1x) are compared to the torques that are actually input (T2x, T1x) at each time step (Appendix B has the program listing). If there are no effects of singularity then the theoretical torque and input torque should be very similar. From Figure 2 the sum of the moments about the center of gravity of link three is:

ΣM3=M3(L(3,2)2(wdx(3))=T2x + Fz2y - Fy2z

so

T2x=M3(L(3,2))2(wdx(3)) - Fz2y + Fy2z=Tory2x

where

y=L(3,1)(cos(θy3))

Fy2=(-M3)(ay3)

z=L(3,1)(cos(θy3))

Fz2=(-M3)(az3)

Sum of the moments about the center of gravity of link two is:

ΣM2=Ixx2(wdx(2))=T1x-T2x + Fz1cos(θy2)(L(2,1))

-Fy1sin(θy2)(L(2,1)) + Fz2cos(θy2)(L(2,2))

-Fy2sin(θy2)(L(2,2))

so

$$T1x = M2(L(2,2))2(wdx(2)) + T2x - Fz1\cos(\theta y2)(L(2,1))$$

$$+ Fy1\sin(\theta y2)(L(2,1)) - Fz2(L(2,2))\cos(\theta y2)$$

$$+ Fy2\sin(\theta y2)(L(2,2)) = Tory1x$$

where

$$Fz1 = Fz2 - M2(az2)$$

$$Fy1 = Fy2 - M2ay2.$$

For comparison the % error is used for difference in torque for joint two and joint one:

$$Errt2x = (Tory2x - T2x)/(maxTory2x) \times 100$$

$$Errt1x = (Tory1x - T1x)/(maxTory1x) \times 100$$

V. RESULTS

A. MOVEMENT OF LINK THREE

Analysis of the movement of only link three shows very good results for program validation. Figure 5 shows a plot of Euler angles for both theoretical (thθy3) and simulated (θy3) values, the graph shows indistinguishable differences. To further visualize the difference Figure 6 was plotted, which is the % error between thθy and θy versus time. There seems to be greater error (0.0032%) at around 0.8 seconds than at 0.2 seconds. This could possibly be caused by error buildup in the computation due to round off error from subroutine LEQT2F and truncation error from approximating the solution to the second order differential equation by the ADAMS method. Additionally, inaccuracies could occur in estimating the value of π and using it in trigonometric calculations. However, the % error is small and is acceptable to verify the proper operation of the program for the single degree of freedom case.

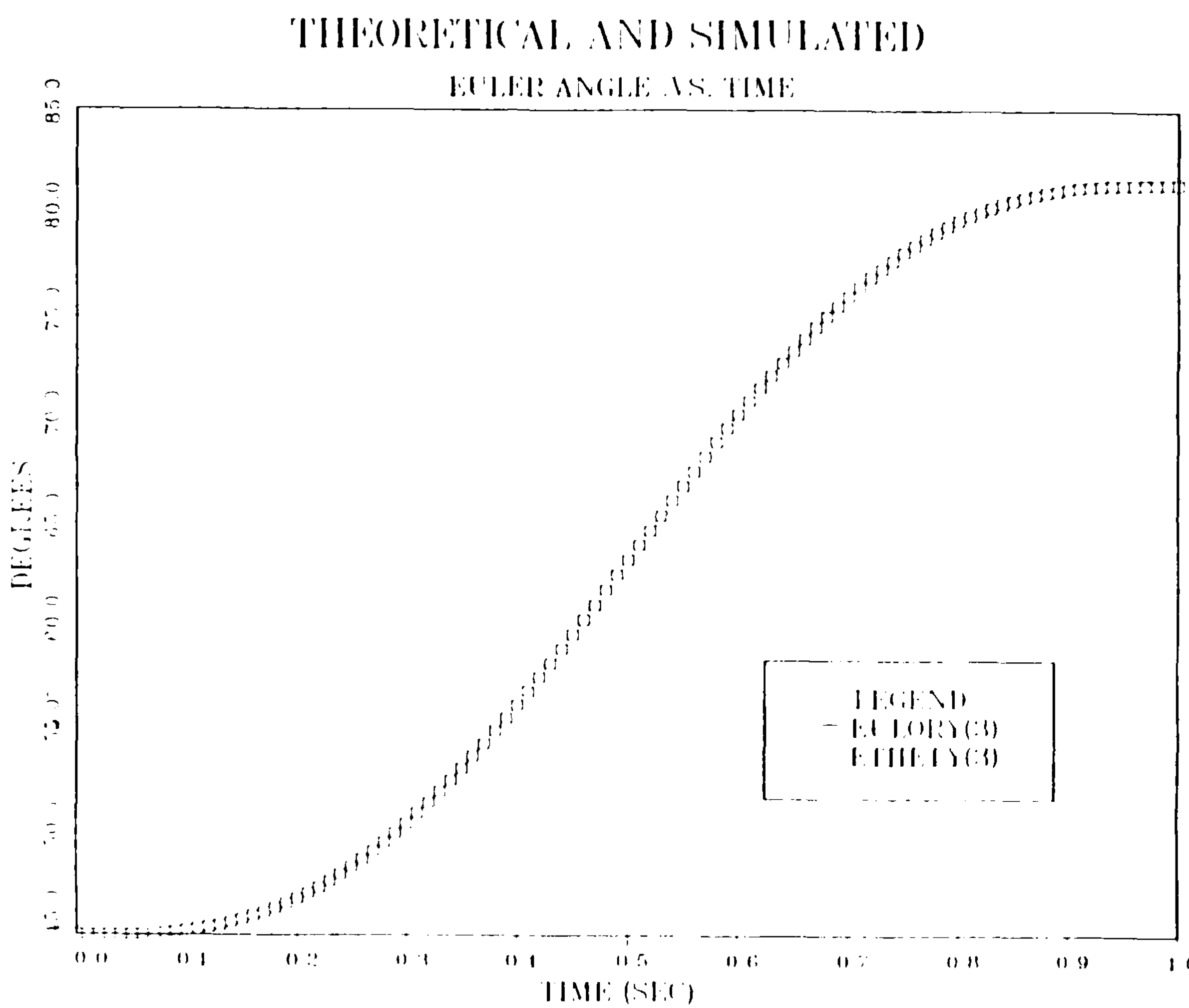


Figure 5. Theoretical and Simulated Euler Angle vs. Time

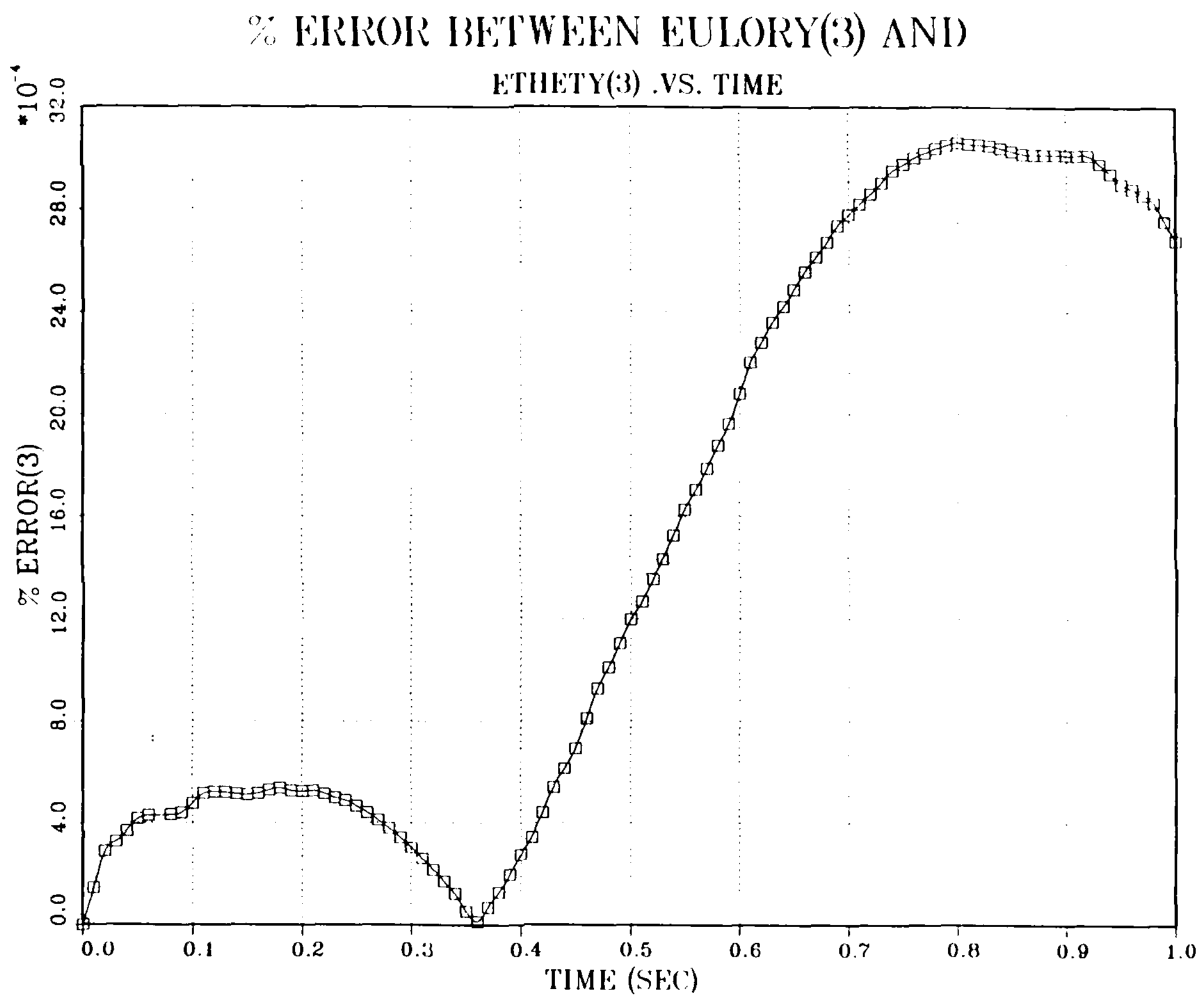


Figure 6. Percent Error Between Theoretical and Simulated Euler Angle vs. Time

B. MOVEMENT OF LINK TWO AND THREE

Analysis of the movement of link two and three is the crucial test of how the simulation deals with the problem of singularity. A torque was input to joint two and an opposite torque to joint one. At some point the alignments of the two links will have some absolute angle of 0° (Figure 7) relative to each other. At this time if singularity exists there is no longer any control of the links and accelerations and velocities vary abnormally, never returning to the level they were at before the singular position was reached [Ref. 8]. So the reason for comparing the values of the computed torques (Tory2x, Tory1x) given the position variables solved for by the simulation program and the torques input to the joints (T2x, T1x), is to check for abnormalities. Figure 8 shows the graph of computed and input torques for joint two and Figure 9 shows the graph of computed and input torques for joint one versus time. The two curves match very well and shows almost no deviation between them for the scale used.

When the % errors are plotted between computed torque and input torque versus time for links two and one (Figures 10 and 11) again very little % error is observed with the largest being around 0.024% at time 2.8 seconds for torque input at joint one. This may be attributed to the similar reasons as the one-link since now both link two and three are moving these errors are building as time increases. It

ETHETY(2) & ETHETY(3) .VS. TIME

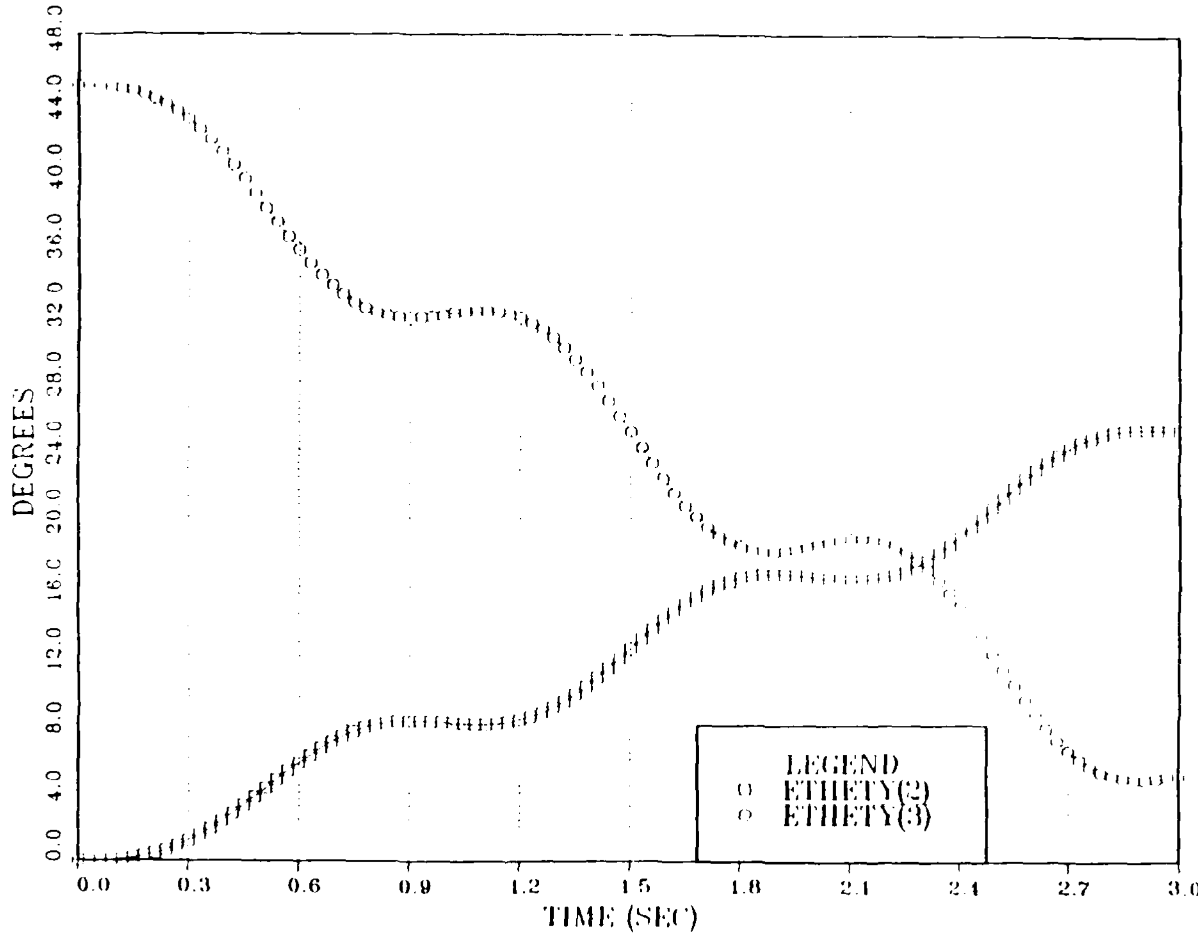


Figure 7. Simulated Movement of Link Two and Three vs. Time

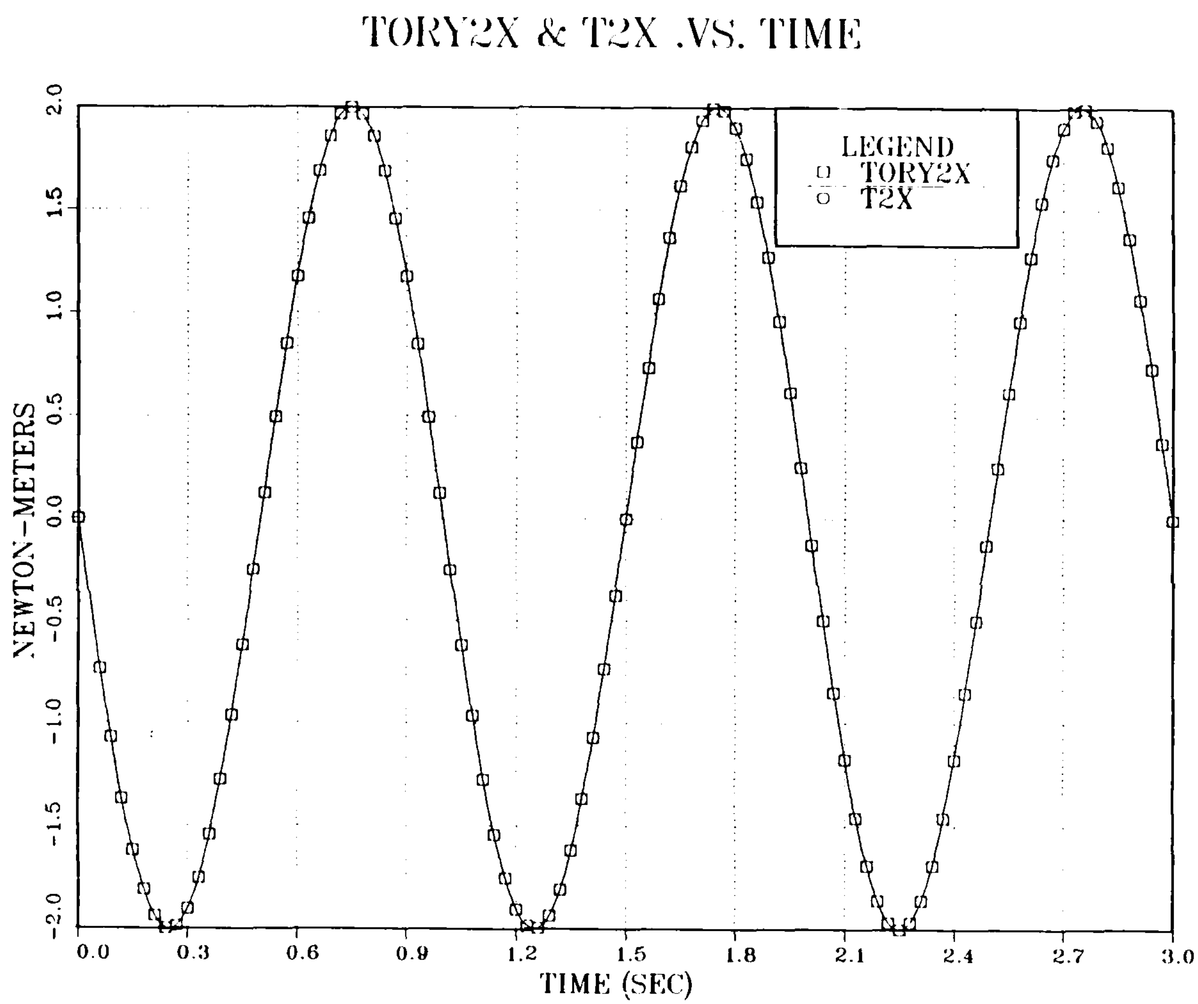


Figure 8. Computed and Input Values of Torques
at Joint Two vs. Time

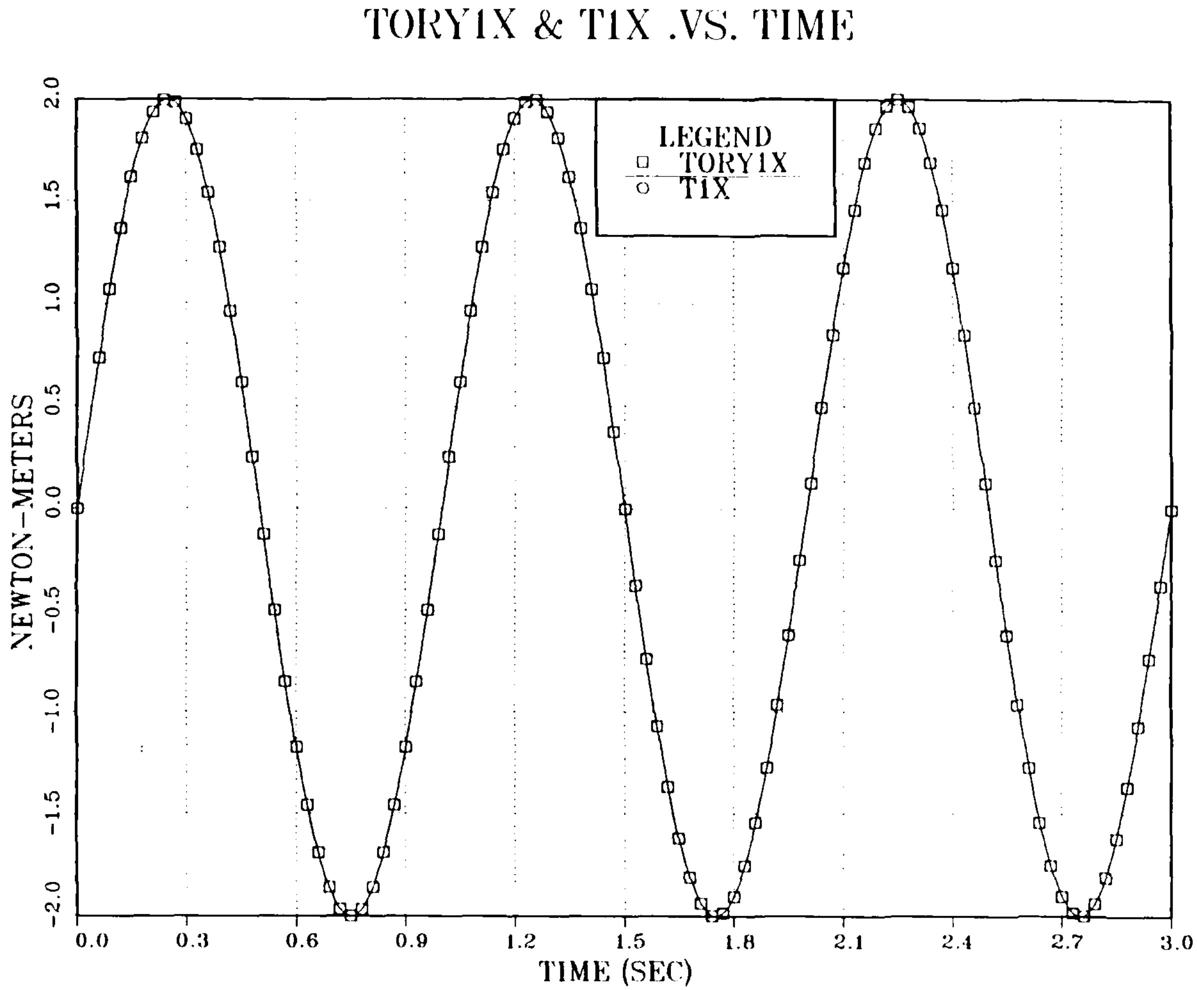


Figure 9. Computed and Input Values of Torques at Joint One vs. Time

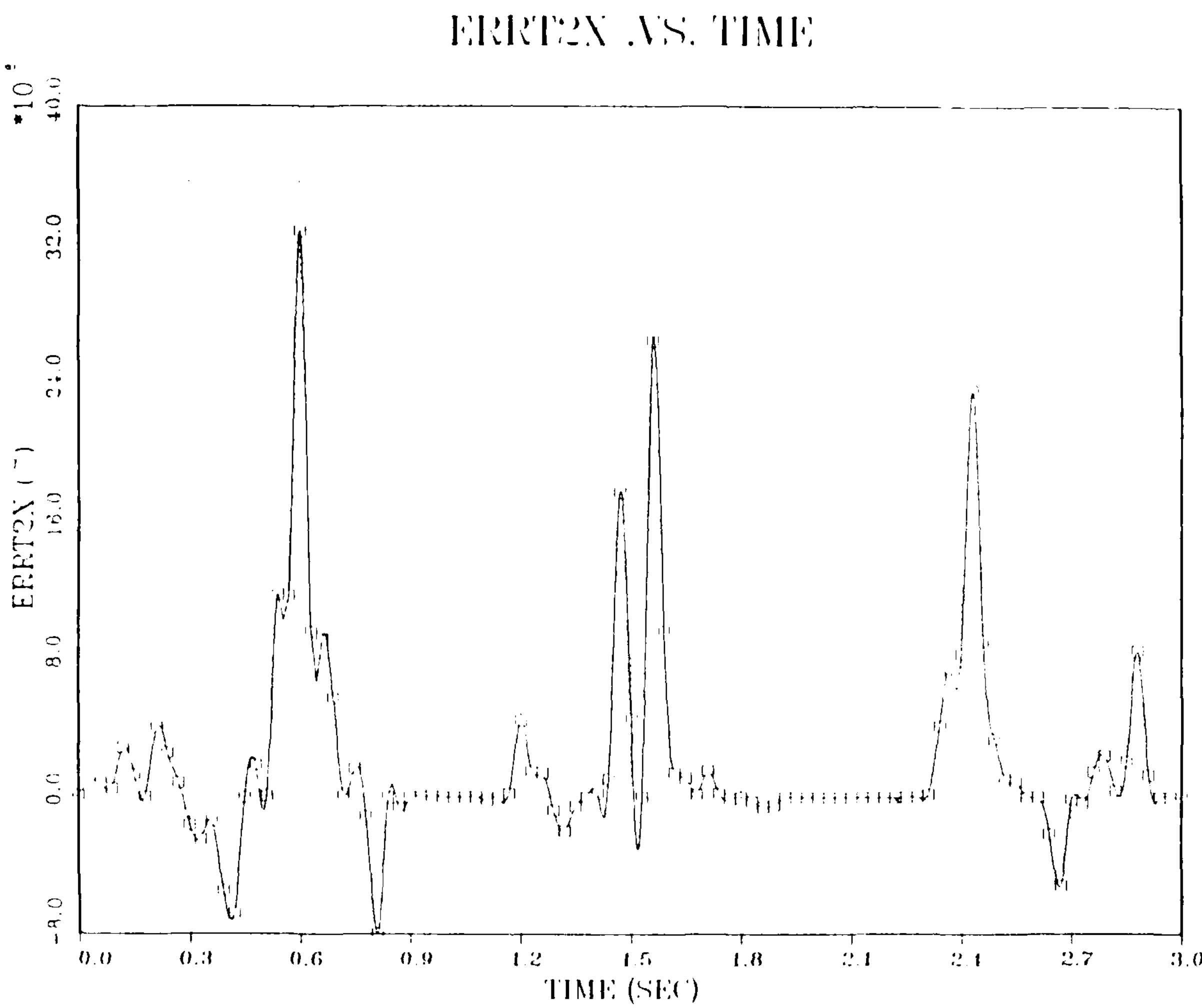


Figure 10. Percent Error Between Tory2x and T2x vs. Time

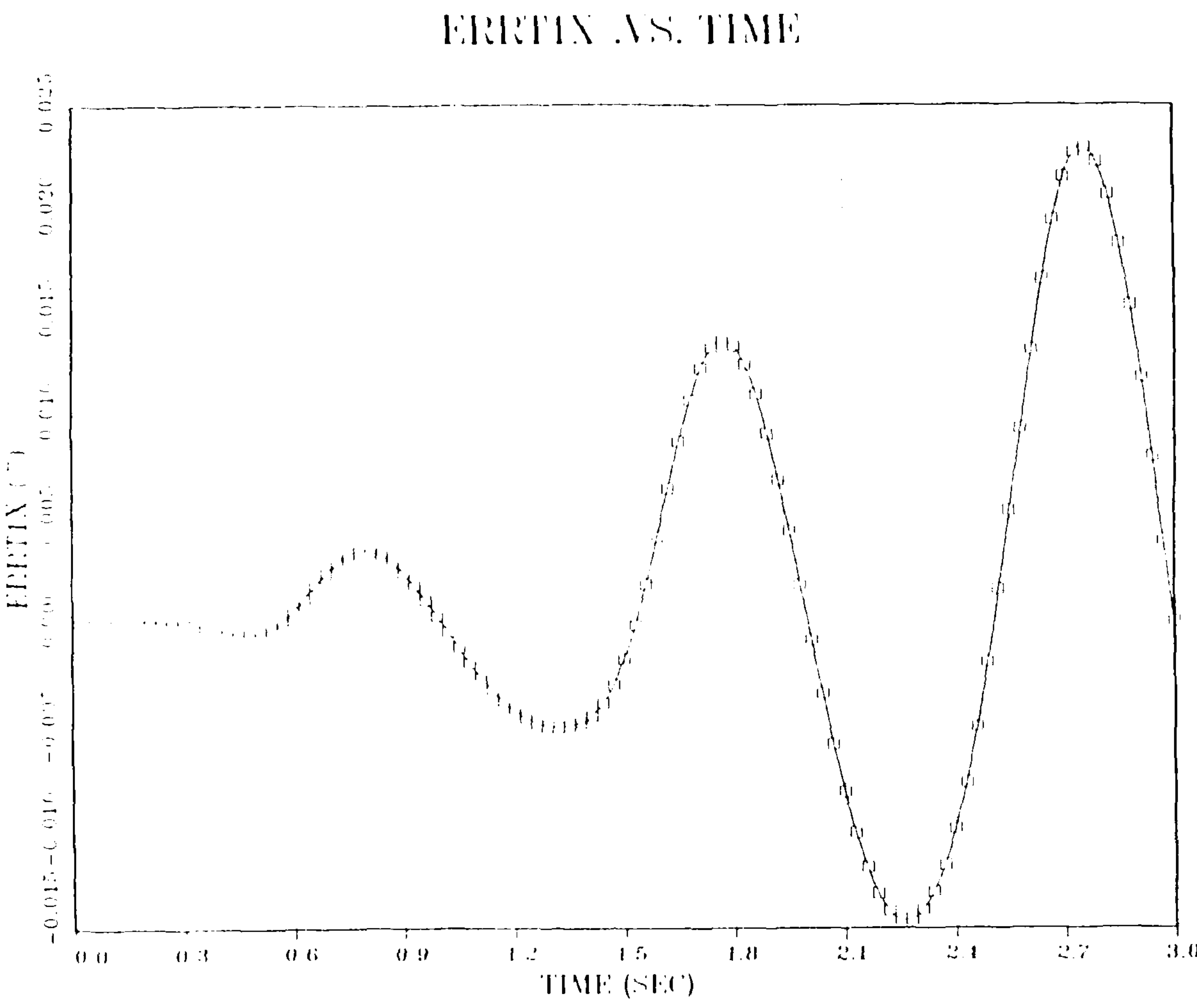


Figure 11. Percent Error Between Torytx and Ttx vs. Time

is also observed in Figure 10 that the % error is not smooth but erratic and causes a "spikey" curve fit. However, the error comes back down to the zero plateau instead of remaining at a high level which is what would have happened had singularity occurred. The overall % errors are small and so lends credability to the simulation model. Figure 7 was plotted to see at what point the two links align themselves and to get a picture of about how long they are close (within one degree) to the point of singularity. It appears to be 0.5 seconds which is enough time for singularity to have a strong effect [Ref. 8].

VI. CONCLUSIONS

The ability of a global two degree of freedom robot arm to maneuver through a point of singularity under applied torques was demonstrated. This was verified by comparing the computed torque at joint one and two in the x direction to the values that were input. There were no unusual or abnormal results occurring in the acceleration or velocities and so little error was produced.

VII. RECOMMENDATIONS

The following recommendations are provided:

1. Develop a linearized manipulator model and a corresponding controller for the two degree of freedom case.
2. Validate the approach via actual empirical tests for the two dimensional case. This will establish the difficulty of determining accurate constants for the simulation and controller design.
3. Demonstrate the model and controller in 3 dimensions with 3 links. The difficulty here arises in analyzing the direction of a given joint torque in 3 dimensions. This could probably be done by finding a unit normal vector perpendicular to the joint in the x, y and z direction and multiplying it by the torque magnitude.
4. Validate the approach by implementation for the three dimensional case.

APPENDIX A

SIMULATION PROGRAM FOR MOVEMENT OF LINK THREE

```
TERMINAL
METHOD ADAMS
PRINT .01,ETHETY(3),EULORY(3),ERROR(3)
CONTROL FINTIM =1.0, DELMAX =.01, DELPRT = .01
SAVE .01,ETHETY(3),EULORY(3),ERROR(3)
GRAPH(DE=TEK618) TIME,ETHETY(3)
GRAPH(DE=TEK618) TIME,ERROR(3)
GRAPH(DE=TEK618) TIME,EULORY(3)
D      DIMENSION MATA(27,27),MASS(3,2),L(3,2),RX(3,2),RY(3,2),RZ(3,2)
D      DIMENSION IXX(3,2),IXZ(3,2),IXY(3,2),IYY(3,2),IYZ(3,2),IZZ(3,2)
D      INTEGER IER,I,RUN,M,N,IA,IDGT
EXCLUDE IA,IDGT,IER,I,RUN,M,N
ARRAY MATB(27),LCOGX(3),LCOGY(3),LCOGZ(3),ETHETX(3),ETHETY(3),ETHETZ(3)
ARRAY CTHETX(3),CTHETY(3),CTHETZ(3),THDDOT(3),IXXA(3),ERROR(3)
ARRAY VECTA0(3),VECTB0(3),VECTA1(3),VECTB1(3),VECTA2(3),VECTB2(3)
ARRAY WDX(3),WDY(3),WDZ(3),WX(3),WY(3),WZ(3),RBG1(3),RAG1(3),THEORY(3)
ARRAY RBG2(3),RAG2(3),RBG3(3),THETXR(3),THETYR(3),THETZR(3),EULORY(3)
ARRAY HDX(2),HDY(2),HDZ(2)
ARRAY SUMHDX(3),SUMHDY(3),SUMHDZ(3),WKAREA(850)
D      DATA MATA/729 * 0./

INITIAL

*    INPUT PARAMETER CONSTANTS
           A = 10.0
           P = 0.0
           W = 2*PI
           IDGT = 4
           G=0.0
           N=27
           M=1
           IA =27
           RUN = 1

*       INPUT JOINT LOCATIONS IN METERS
            JX0 = 0.0
            JY0 = 0.0
            JZ0 = 0.0
            JX1 = 0.0
            JY1 = 1.0
            JZ1 = 0.0
            JX2 = 0.0
            JY2 = 2.0
            JZ2 = 0.0

*       INPUT TORQUE CONSTANTS
            T0X = 0.0
            T0Y = 0.0
            T0Z = 0.0
            T1X = 0.0
            T1Y = 0.0
            T1Z = 0.0
            T2Y = 0.0
            T2Z = 0.0

*INPUT DISTANCE FROM CENTER OF LINK TO CENTER OF MASS FOR EACH LINK ENDS
            L(1,1) = 0.50
            L(1,2) = 0.50
            L(2,1) = 0.50
            L(2,2) = 0.50
            L(3,1) = 0.50
```

```
      L(3,2) = 0.50

*     INPUT MASS AT LINK ENDS IN KILOGRAMS
      MASS(1,1) = 2.5
      MASS(1,2) = 2.5
      MASS(2,1) = 2.5
      MASS(2,2) = 2.5
      MASS(3,1) = 2.5
      MASS(3,2) = 2.5

*     INPUT OMEGA AND OMEGA DOT
       DO 30 I = 1,3
          WX(I) = 0.0
          WY(I) = 0.0
          WZ(I) = 0.0
          WDX(I) = 0.0
          WDY(I) = 0.0
          WDZ(I) = 0.0
30      CONTINUE

*     INPUT INITIAL VALUES OF EULER ANGLE THETA AND CONVERT TO RADIANS
      ETHETX(1) = 90.0
       TX1 = ETHETX(1) * DEGRA
       ETHETY(1) = 0.0
       TY1 = ETHETY(1) * DEGRA
       ETHETZ(1) = 90.0
       TZ1 = ETHETZ(1) * DEGRA
       ETHETX(2) = 90.0
       TX2 = ETHETX(2) * DEGRA
       ETHETY(2) = 0.0
       TY2= ETHETY(2) * DEGRA
       ETHETZ(2) = 90.0
       TZ2 = ETHETZ(2) * DEGRA
       ETHETX(3) = 90.0
       TX3 = ETHETX(3) * DEGRA
       ETHETY(3) = 45.0
       TY3 = ETHETY(3) * DEGRA
       ETHETZ(3) = 45.0
       TZ3 = ETHETZ(3) * DEGRA

*     INPUT LOCATION OF LINK CENTERS OF GRAVITY
      LCOGX(1) = 0.0
      X1 = LCOGX(1)
      LCOGY(1) = 0.5
      Y1 = LCOGY(1)
      LCOGZ(1) = 0.0
      Z1 = LCOGZ(1)
      LCOGX(2) = 0.0
      X2 = LCOGX(2)
      LCOGY(2) = 1.5
      Y2 = LCOGY(2)
      LCOGZ(2) = 0.0
      Z2 = LCOGZ(2)
      LCOGX(3) = 0.0
      X3 = LCOGX(3)
      THERAD = ETHETY(3) * DEGRA
      LCOGY(3) = 2.0 + COS(THERAD) * L(3,1)
      Y3 = LCOGY(3)
      LCOGZ(3) = L(3,1) * SIN(THERAD)
      Z3 = LCOGZ(3)

*     INPUT MASS OF EACH LINK IN KG AND COMPUTE WEIGHTS IN NEWTONS
       MASS1 = 5.0
       MASS2 = 5.0
       MASS3 = 5.0
       W1 = MASS1*G
       W2 = MASS2*G
       W3 = MASS3*G
```

```
*    INPUT ACCELERATION OF JOINT ZERO
                AOX = 0.0
                AOY = 0.0
                AOZ = 0.0

DERIVATIVE

NOSORT
*       INPUT JOINT EQUATIONS

*       INITIALIZE MATRIX B TO ZERO
                DO 10 I = 1,27
                    MATB(I) = 0.0
10          CONTINUE

*       INPUT TORQUE AT JOINTS
                    T2X = A*SIN (W*TIME +P)

*           JOINT ZERO  AB = AG1 + (WD1 X RB/G1) + W1 X (W1 X RB/G1)
                    VECTA0(1) = WDX(1)
                    VECTA0(2) = WDY(1)
                    VECTA0(3) = WDZ(1)
                    RBG1(1) = JX0 - LCOGX(1)
                    RBG1(2) = JY0 - LCOGY(1)
                    RBG1(3) = JZ0 - LCOGZ(1)
                CALL CPROD(VECTA0,RBG1,MIA0,MJA0,MKA0)
                    VECTA0(1) = WX(1)
                    VECTA0(2) = WY(1)
                    VECTA0(3) = WZ(1)
                CALL CPROD(VECTA0,RBG1,MIB0,MJB0,MKB0)
                    VECTB0(1) = MIB0
                    VECTB0(2) = MJB0
                    VECTB0(3) = MKB0
                CALL CPROD(VECTA0,VECTB0,MIC0,MJC0,MKC0)

*   JOINT ONE EQUATIONS--- AA = AG1 + (WD1 X RA/G1) + W1 X (W1 X RA/G1)
                    VECTA1(1) = WDX(1)
                    VECTA1(2) = WDY(1)
                    VECTA1(3) = WDZ(1)
                    RAG1(1) = JX1 - LCOGX(1)
                    RAG1(2) = JY1 - LCOGY(1)
                    RAG1(3) = JZ1 - LCOGZ(1)
                CALL CPROD(VECTA1,RAG1,MIA1,MJA1,MKA1)
                    VECTA1(1) = WX(1)
                    VECTA1(2) = WY(1)
                    VECTA1(3) = WZ(1)
                CALL CPROD (VECTA1,RAG1,MIB1,MJB1,MKB1)
                    VECTB1(1) = MIB1
                    VECTB1(2) = MJB1
                    VECTB1(3) = MKB1
                CALL CPROD (VECTA1,VECTB1,MIC1,MJC1,MKC1)

*     AB = AG2 + (WD2 X RB/G2) + W2 X (W2 X RB/G2)
                    VECTA1(1) = WDX(2)
                    VECTA1(2) = WDY(2)
                    VECTA1(3) = WDZ(2)
                    RBG2(1) = JX1 - LCOGX(2)
                    RBG2(2) = JY1 - LCOGY(2)
                    RBG2(3) = JZ1 - LCOGZ(2)
                 CALL CPROD (VECTA1,RBG2,MIA2,MJA2,MKA2)
                    VECTA1(1) = WX(2)
                    VECTA1(2) = WY(2)
                    VECTA1(3) = WZ(2)
                 CALL CPROD (VECTA1,RBG2,MIB2,MJB2,MKB2)
                    VECTB1(1) = MIB2
                    VECTB1(2) = MJB2
                    VECTB1(3) = MKB2
                 CALL CPROD (VECTA1,VECTB1,MIC2,MJC2,MKC2)
```

```
* JOINT TWO EQUATIONS
*       AA = AG2 + (WD2 X RA/G2) + W2 X (W2 X RA/G2)
                  VECTA2(1) = WDX(2)
                  VECTA2(2) = WDY(2)
                  VECTA2(3) = WDZ(2)
                  RAG2(1) = JX2 - LCOGX(2)
                  RAG2(2) = JY2 - LCOGY(2)
                  RAG2(3) = JZ2 - LCOGZ(2)
               CALL CPROD (VECTA2,RAG2,MIA3,MJA3,MKA3)
                  VECTA2(1) = WX(2)
                  VECTA2(2) = WY(2)
                  VECTA2(3) = WZ(2)
               CALL CPROD (VECTA2,RAG2,MIB3,MJB3,MKB3)
                  VECTB2(1) = MIB3
                  VECTB2(2) = MJB3
                  VECTB2(3) = MKB3
               CALL CPROD(VECTA2,VECTB2,MIC3,MJC3,MKC3)

*       AB = AG3 + (WD3 X RB/G3) + W3 X (W3 X RB/G3)
                  VECTA2(1) = WDX(3)
                  VECTA2(2) = WDY(3)
                  VECTA2(3) = WDZ(3)
                  RBG3(1) = JX2 - LCOGX(3)
                  RBG3(2) = JY2 - LCOGY(3)
                  RBG3(3) = JZ2 - LCOGZ(3)
               CALL CPROD (VECTA2,RBG3,MIA4,MKA4,MKA4)
                  VECTA2(1) = WX(3)
                  VECTA2(2) = WY(3)
                  VECTA2(3) = WZ(3)
               CALL CPROD (VECTA2,RBG3,MIB4,MJB4,MKB4)
                  VECTB2(1) = MIB4
                  VECTB2(2) = MJB4
                  VECTB2(3) = MKB4
               CALL CPROD (VECTA2,VECTB2,MIC4,MJC4,MKC4)

*     SUM OF MOMENTS EQUATIONS
*         CONVERT EULER ANGLES FROM DEGREES TO RADIANS
            DO 40 I = 1,3
              THETXR(I) = ETHETX(I) * DEGRA
              THETYR(I) = ETHETY(I) * DEGRA
              THETZR(I) = ETHETZ(I) * DEGRA

*       COMPUTE HX DOT,HY DOT,HZ DOT
                  RX(I,1) = -L(I,1) * COS(THETXR(I))
                  RX(I,2) = L(I,2) * COS(THETXR(I))
                  RY(I,1) = -L(I,1) * COS(THETYR(I))
                  RY(I,2) = L(I,2) * COS(THETYR(I))
                  RZ(I,1) = -L(I,1) * COS(THETZR(I))
                  RZ(I,2) = L(I,2) * COS(THETZR(I))
             IXX(I,1)=MASS(I,1)*((RY(I,1)*RY(I,1))+(RZ(I,1)*RZ(I,1)))
             IXX(I,2)=MASS(I,2)*((RY(I,2)*RY(I,2))+(RZ(I,2)*RZ(I,2)))
             IXZ(I,1)=MASS(I,1) * RZ(I,1) * RX(I,1)
             IXZ(I,2)=MASS(I,2) * RZ(I,2) * RX(I,2)
             IXY(I,1)=MASS(I,1) * RX(I,1) * RY(I,1)
             IXY(I,2)=MASS(I,2) * RX(I,2) * RY(I,2)
                HDX(1) = WDX(1)*IXX(I,1)-WDZ(I)*IXZ(I,1)-WDY(I)*IXY(I,1)
                HDX(2) = WDX(2)*IXX(I,2)-WDZ(I)*IXZ(I,2)-WDY(I)*IXY(I,2)
             IYY(I,1) = MASS(I,1) * ((RX(I,1)*RX(I,1)) + (RZ(I,1)*RZ(I,1)))
             IYY(I,2) = MASS(I,2) * ((RX(I,2)*RX(I,2)) + (RZ(I,2)*RZ(I,2)))
             IYZ(I,1) = MASS(I,1) * RY(I,1) * RZ(I,1)
             IYZ(I,2) = MASS(I,2) * RY(I,2) * RZ(I,2)
                HDY(1) = WDY(I)*IYY(I,1)-WDX(I)*IXY(I,1)-WDZ(I)*IYZ(I,1)
                HDY(2) = WDY(I)*IYY(I,2)-WDX(I)*IXY(I,2)-WDZ(I)*IYZ(I,2)
             IZZ(I,1) = MASS(I,1) * ((RX(I,1)*RX(I,1)) + (RY(I,1)*RY(I,1)))
             IZZ(I,2) = MASS(I,2) * ((RX(I,2)*RX(I,2)) + (RY(I,2)*RY(I,2)))
                HDZ(1) = WDZ(I)*IZZ(I,1)-WDX(I)*IXZ(I,1)-WDY(I)*IYZ(I,1)
                HDZ(2) = WDZ(I)*IZZ(I,2)-WDX(I)*IXZ(I,2)-WDY(I)*IYZ(I,2)
             IXXA(I)=MASS(I,2)*((L(I,2)+L(I,1))**2)
             SUMHDX(I) = HDX(1) + HDX(2)
```

```
          SUMHDY(I) = HDY(1) + HDY(2)
          SUMHDZ(I) = HDZ(1) + HDZ(2)
40      CONTINUE

*   TEST TO SEE WHICH CONSTRAINT IS IN EFFECT 1,2 OR 3
          IF (RUN .EQ. 1) GO TO 1
          IF (RUN .EQ. 2) GO TO 2
          IF (RUN .EQ. 3) GO TO 3

*   INITIAIIZE MATRIX ACCORDING TO CONSTRAINT
1       DO 60 I = 1,18
              MATA(I,I) = 1.0
60      CONTINUE
          GO TO 4
2       DO 70 I = 1,9
              MATA(I,I) = 1.0
70      CONTINUE
          GO TO 7

*   ENTER CONSTANTS INTO MATRIX A
*    LINK ONE
*      SUM OF FORCES IN THE X DIRECTION
3       MATA(1,1) = 1.0
          MATA(1,4) = MASS1
          MATA(1,10) = -1.0

*      SUM OF FORCES IN Y DIRECTION
          MATA(2,2) = 1.0
          MATA(2,5) = MASS1
          MATA(2,11) = -1.0

*     SUM OF FORCES IN Z DIRECTION
          MATA(3,3) = 1.0
          MATA(3,6) = MASS1
          MATA(3,12) = -1.0

*      SUM OF FORCES LINK ONE EQUAL
          MATB(3) = -W1

*    EQUATIONS AT JOINT ZERO
*      IN THE X DIRECTION
          MATA(4,4) = 1.0
          MATA(4,8) = RBG1(3)
          MATA(4,9) = -RBG1(2)

          MATB(4)= A0X - MIC0

*      IN THE Y DIRECTION
          MATA(5,5) = 1.0
          MATA(5,7) = -RBG1(3)
          MATA(5,9) = RBG1(1)

          MATB(5) = A0Y - MJC0

*      IN THE Z DIRECTION
          MATA(6,6) = 1.0
          MATA(6,7) = RBG1(2)
          MATA(6,8) = -RBG1(1)

          MATB(6) = A0Z - MKC0

*   SUM OF MOMENTS EQUATIONS FOR LINK ONE IN THE X,Y,Z DIRECTIONS

          MATA(7,2) = RBG1(3)
          MATA(7,3) = -RBG1(2)
          MATA(7,7) = -(IXX(1,1) + IXX(1,2))
          MATA(7,8) = IXY(1,1) + IXY(1,2)
          MATA(7,9) = IXZ(1,1) + IXZ(1,2)
          MATA(7,11) = -RAG1(3)
```

```
            MATA(7,12) = RAG1(2)

            MATB(7) = T1X - T0X

            MATA(8,1) = -RBG1(3)
            MATA(8,3)= RBG1(1)
            MATA(8,7) = IXY(1,1) + IXY(1,2)
            MATA(8,8) = -(IYY(1,1) + IYY(1,2))
            MATA(8,9) = IYZ(1,1) + IYZ(1,2)
            MATA(8,10) = RAG1(3)
            MATA(8,12) = -RAG1(1)

            MATB(8) = T1Y - T0Y

            MATA(9,1) = RBG1(2)
            MATA(9,2) = -RBG1(1)
            MATA(9,7) = IXZ(1,1) + IXZ(1,2)
            MATA(9,8) = IYZ(1,1) + IYZ(1,2)
            MATA(9,9) = -(IZZ(1,1) + IZZ(1,2))
            MATA(9,10) = -RAG1(2)
            MATA(9,11) = RAG1(1)

            MATB(9) = T1Z - T0Z

*  LINK TWO
*     SUM OF FORCES IN X DIRECTION
7       MATA(10,10) = 1.0
            MATA(10,13) = MASS2
            MATA(10,19) = -1.0

*     SUM OF FORCES IN THE Y DIRECTION
            MATA(11,11) = 1.0
            MATA(11,14) = MASS2
            MATA(11,20) = -1.0

*     SUM OF FORCES IN THE Z DIRECTION
            MATA(12,12) = 1.0
            MATA(12,15) = MASS2
            MATA(12,21) = -1.0

*     SUM OF FORCES LINK TWO EQUAL
           MATB(12) = -W2

*    EQUATIONS AT JOINT ONE
*      IN THE X DIRECTION
            MATA(13,4) = -1.0
            MATA(13,8) = -RAG1(3)
            MATA(13,9) = RAG1(2)
            MATA(13,13) = 1.0
            MATA(13,17) = RBG2(3)
            MATA(13,18) = -RBG2(2)

            MATB(13) = MIC1 - MIC2

*      IN THE Y DIRECTION
            MATA(14,5) = -1.0
            MATA(14,7) = RAG1(3)
            MATA(14,9) = -RAG1(1)
            MATA(14,14) = 1.0
            MATA(14,16) = -RBG2(3)
            MATA(14,18) = RBG2(1)

            MATB(14) = MJC1 - MJC2

*      IN THE Z DIRECTION
            MATA(15,6) = -1.0
            MATA(15,7) = -RAG1(2)
            MATA(15,8) = RAG1(1)
            MATA(15,15) = 1.0
```

```
          MATA(15,16) = RBG2(2)
          MATA(15,17) = -RBG2(1)

          MATB(15) = MKC1 - MKC2

*   SUM OF MOMENTS EQUATIONS FOR LINK TWO IN THE X,Y,Z DIRECTIONS
          MATA(16,11) = RBG2(3)
          MATA(16,12) = -RBG2(2)
          MATA(16,16) = -(IXX(2,1) + IXX(2,2))
          MATA(16,17) = IXY(2,1) + IXY(2,2)
          MATA(16,18) = IXZ(2,1) + IXZ(2,2)
          MATA(16,20) = -RAG2(3)
          MATA(16,21) = RAG2(2)
          MATB(16) = -T1X + T2X
          IF(RUN .EQ. 2) GO TO 11

          MATA(17,10) = -RBG2(3)
          MATA(17,12) = RBG2(1)
          MATA(17,16) = IXY(2,1) + IXY(2,2)
          MATA(17,17) = -(IYY(2,1) + IYY(2,2))
          MATA(17,18) = IYZ(2,1) + IYZ(2,2)
          MATA(17,19) = RAG2(3)
          MATA(17,21) = -RAG2(1)

          MATB(17) =  - T1Y + T2Y

          MATA(18,10) = RBG2(2)
          MATA(18,11) = -RBG2(1)
          MATA(18,16) = IXZ(2,1) + IXZ(2,2)
          MATA(18,17) = IYZ(2,1) + IYZ(2,2)
          MATA(18,18) = -(IZZ(2,1) + IZZ(2,2))
          MATA(18,19) = -RAG2(2)
          MATA(18,20) = RAG2(1)

          MATB(18) =  - T1Z + T2Z

          IF (RUN .EQ. 3) GO TO 4
11       MATA(17,17) = 1.0
          MATA(18,18) = 1.0

*   LINK THREE
*      SUM OF FORCES IN THE X DIRECTION
4        MATA(19,19) = 1.0
          MATA(19,22) = MASS3

*      SUM OF FORCES IN THE Y DIRECTION
          MATA(20,20) = 1.0
          MATA(20,23) = MASS3

*      SUM OF FORCES IN THE Z DIRECTION
          MATA(21,21) = 1.0
          MATA(21,24) = MASS3

          MATB(21) = -W3

*   EQUATIONS AT JOINT TWO
*      IN THE X DIRECTION
          MATA(22,13) = -1.0
          MATA(22,17) = -RAG2(3)
          MATA(22,18) = RAG2(2)
          MATA(22,22) = 1.0
          MATA(22,26) = RBG3(3)
          MATA(22,27) = -RBG3(2)

          MATB(22) = MIC3 - MIC4

*      IN THE Y DIRECTION
          MATA(23,14) = -1.0
          MATA(23,16) = RAG2(3)
```

```
        MATA(23,18) = -RAG2(1)
        MATA(23,23) = 1.0
        MATA(23,25) = -RBG3(3)
        MATA(23,27) = RBG3(1)

        MATB(23) = MJC3 - MJC4

*     IN THE Z DIRECTION
        MATA(24,15) = -1.0
        MATA(24,16) = -RAG2(2)
        MATA(24,17) = RAG2(1)
        MATA(24,24) = 1.0
        MATA(24,25) = RBG3(2)
        MATA(24,26) = -RBG3(1)

        MATB(24) = MKC3 - MKC4

*    SUM OF MOMENTS EQUATIONS FOR LINK THREE IN THE X,Y,Z DIRECTIONS
        MATA(25,20) = RBG3(3)
        MATA(25,21) = -RBG3(2)
        MATA(25,25) = -(IXX(3,1) + IXX(3,2))
        MATA(25,26) = IXY(3,1) + IXY(3,2)
        MATA(25,27) = IXZ(3,1) + IXZ(3,2)
        MATB(25) = - T2X
        IF(RUN .EQ. 1 .OR. RUN .EQ. 2) GO TO 12

        MATA(26,19) = -RBG3(3)
        MATA(26,21) = RBG3(1)
        MATA(26,25) = IXY(3,1) + IXY(3,2)
        MATA(26,26) = -(IYY(3,1) + IYY(3,2))
        MATA(26,27) = IYZ(3,1) + IYZ(3,2)

        MATB(26) = - T2Y

        MATA(27,19) = RBG3(2)
        MATA(27,20) = -RBG3(1)
        MATA(27,25) = IXZ(3,1) + IXZ(3,2)
        MATA(27,26) = IYZ(3,1) + IYZ(3,2)
        MATA(27,27) = -(IZZ(3,1) + IZZ(3,2))

       MATB(27) = - T2Z

        IF (RUN .EQ. 3) GO TO 13
12      MATA(26,26) = 1.0
        MATA(27,27) = 1.0

* CALL EQUATION SOLVER PROGRAM FROM IMSL
13    CALL LEQT2F(MATA,M,N,IA,MATB,IDGT,WKAREA,IER)
       IF (IER .NE. 0) CALL ENDJOB

* FIND LCOGX,LCOGY,LCOGZ,THETA VALUES,WX,WY,WZ
       IF(RUN .EQ. 1) GO TO 6
       IF (RUN .EQ. 2) GO TO 9

*     LINK ONE
        AX1 = MATB(4)
        VELX1 = INTGRL(0,AX1)
        LCOGX1 = INTGRL(X1,VELX1)
        LCOGX(1) = LCOGX1
        AY1 = MATB(5)
        VELY1 = INTGRL(0,AY1)
        LCOGY1 = INTGRL(Y1,VELY1)
        LCOGY(1) = LCOGY1
        AZ1 = MATB(6)
        VELZ1 = INTGRL(0,AZ1)
        LCOGZ1 = INTGRL(Z1,VELZ1)
        LCOGZ(1) = LCOGZ1
        WD1X = MATB(7)
```

```
         W1X = INTGRL(0,WD1X)
         THEXR1 = INTGRL(TY1,W1X)
         JX0= LCOGX(1) - L(1,1) * COS(TX1)
         WDX(1) = WD1X
         WX(1) = W1X
         CTHETX(1) = THEXR1 * RADEG
         ETHETY(1) = CTHETX(1)
         WD1Y = MATB(8)
         W1Y = INTGRL(0,WD1Y)
         THEYR1 = INTGRL(0.,W1Y)
         JY0 = LCOGY(1) - L(1,1) * COS(THEXR1)
         WDY(1) = WD1Y
         WY(1) = W1Y
         CTHETY(1) = THEYR1 * RADEG
         WD1Z = MATB(9)
         W1Z = INTGRL(0,WD1Z)
         THEZR1 = INTGRL(0.,W1Z)
         WDZ(1) = WD1Z
         WZ(1) = W1Z
         CTHETZ(1) = THEZR1 * RADEG
         ETHETZ(1) = 90.0 - CTHETX(1)
         ETHEZ1 = ETHETZ(1) * DEGRA
         JZ0 = LCOGZ(1) - L(1,1) * COS(ETHEZ1)

*     LINK TWO
9       AX2 = MATB(13)
         VELX2 = INTGRL(0,AX2)
         LCOGX2 = INTGRL(X2,VELX2)
         LCOGX(2) = LCOGX2
         AY2 = MATB(14)
         VELY2 = INTGRL(0,AY2)
         LCOGY2 = INTGRL(Y2,VELY2)
         LCOGY(2) = LCOGY2
         AZ2 = MATB(15)
         VELZ2 = INTGRL(0,AZ2)
         LCOGZ2 = INTGRL(Z2,VELZ2)
         LCOGZ(2) = LCOGZ2
         WD2X = MATB(16)
         W2X = INTGRL(0,WD2X)
         THEXR2 = INTGRL(TY2,W2X)
         JX1 = LCOGX(2) - L(2,1) * COS(TX2)
         WDX(2) = WD2X
         WX(2) = W2X
         CTHETX(2) = THEXR2 * RADEG
         ETHETY(2) = CTHETX(2)
         WD2Y = MATB(17)
         W2Y = INTRGL(0,WD2Y)
         THEYR2 = INTGRL(0.,W2Y)
         JY1 = LCOGY(2) - L(2,1) * COS(THEXR2)
         WDY(2) = WD2Y
         WY(2) = W2Y
         CTHETY(2) =THEYR2 * RADEG
         WD2Z = MATB(18)
         W2Z = INTGRL(0,WD2Z)
         THEZR2 = INTGRL(0.,W2Z)
         WDZ(2) = WD2Z
         WZ(2) = W2Z
         CTHETZ(2) = THEZR2 * RADEG
         ETHETZ(2) = 90.0 - CTHETX(2)
         ETHEZ2 = ETHETZ(2) * DEGRA
         JZ1 = LCOGZ(2) - L(2,1) * COS(ETHEZ2)

*     LINK THREE
6       AX3 = MATB(22)
         VELX3 = INTGRL(0.,AX3)
         LCOGX3 = INTGRL(X3,VELX3)
         LCOGX(3) = LCOGX3
         AY3 = MATB(23)
         VELY3 = INTGRL(0.,AY3)
```

```
          LCOGY3 = INTGRL(Y3,VELY3)
          LCOGY(3) = LCOGY3
          AZ3 = MATB(24)
          VELZ3 = INTGRL(0.,AZ3)
          LCOGZ3 = INTGRL(Z3,VELZ3)
          LCOGZ(3) = LCOGZ3
          WD3X = MATB(25)
          W3X = INTGRL(0.,WD3X)
          THEXR3 = INTGRL(TY3,W3X)
          JX2 = LCOGX(3) - L(3,1) * COS(TX3)
          WDX(3) = WD3X
          WX(3) = W3X
          CTHETX(3) = THEXR3 * RADEG
          ETHETY(3) = CTHETX(3)
          WD3Y = MATB(26)
          W3Y = INTGRL(0.,WD3Y)
          THEYR3 = INTGRL(0.,W3Y)
          JY2 = LCOGY(3) - L(3,1) * COS(THEXR3)
          WDY(3) = WD3Y
          WY(3) = W3Y
          CTHETY(3) = THEYR3 * RADEG
          WD3Z = MATB(27)
          W3Z = INTGRL(0.,WD3Z)
          THEZR3 = INTGRL(0.,W3Z)
          WDZ(3) = WD3Z
          WZ(3) = W3Z
          CTHETZ(3) = THEZR3 * RADEG
          ETHETZ(3) = 90.0 - CTHETX(3)
          ETHEZ3 = ETHETZ(3) * DEGRA
          JZ2 = LCOGZ(3) - L(3,1) * COS(ETHEZ3)
DYNAMIC

      THEORY(3)=((((-2.5/(PI*PI))*SIN(W*TIME))+(5.*TIME)/PI)/IXXA(3))...
        + PI/4.
         EULORY(3) = THEORY(3) * RADEG
          THDDOT(3) = T2X/IXXA(3)
          ERROR(3) = ((ABS(EULORY(3)-ETHETY(3)))/ 81.476)*100.

END
STOP
FORTRAN

*     SUBROUTINE TO COMPUTE THE CROSS PRODUCT OF TWO VECTOR

          SUBROUTINE CPROD(VECTA,VECTB,MI,MJ,MK)
              IMPLICIT REAL*8 (A-Z)
              DIMENSION VECTA(3),VECTB(3)
              MI = VECTA(2) * VECTB(3) - VECTA(3) * VECTB(2)
              MJ = VECTA(3) * VECTB(1) - VECTA(1) * VECTB(3)
              MK = VECTA(1) * VECTB(2) - VECTA(2) * VECTB(1)
           RETURN
           END
```

APPENDIX B

SIMULATION PROGRAM FOR MOVEMENT OF LINK TWO AND THREE

```
TERMINAL
METHOD ADAMS
PRINT .03,T1X,TORY1X,T2X,TORY2X,ERRT2X,ERRT1X,ETHETY(2-3)
CONTROL FINTIM =3.0, DELMAX =.01, DELPRT = .03
SAVE .01,ERRT2X,ERRT1X,TORY1X,TORY2X,T1X,T2X,ETHETY(2),ETHETY(3)
GRAPH(DE=TEK618) TIME,T1X,TORY1X
GRAPH(DE=TEK618) TIME,T2X,TORY2X
GRAPH(DE=TEK618) TIME,ERRT2X
GRAPH(DE=TEK618) TIME,ERRT1X
GRAPH(DE=TEK618) TIME,ETHETY(3),ETHETY(2)
D      DIMENSION MATA(27,27),MASS(3,2),L(3,2),RX(3,2),RY(3,2),RZ(3,2)
D      DIMENSION IXX(3,2),IXZ(3,2),IXY(3,2),IYY(3,2),IYZ(3,2),IZZ(3,2)
D      INTEGER IER,I,RUN,M,N,IA,IDGT
EXCLUDE IA,IDGT,IER,I,RUN,M,N
ARRAY MATB(27),LCOGX(3),LCOGY(3),LCOGZ(3),ETHETX(3),ETHETY(3),ETHETZ(3)
ARRAY CTHETX(3),CTHETY(3),CTHETZ(3)
ARRAY VECTA0(3),VECTB0(3),VECTA1(3),VECTB1(3),VECTA2(3),VECTB2(3)
ARRAY WDX(3),WDY(3),WDZ(3),WX(3),WY(3),WZ(3),RBG1(3),RAG1(3)
ARRAY RBG2(3),RAG2(3),RBG3(3),THETXR(3),THETYR(3),THETZR(3)
ARRAY SUMHDX(3),SUMHDY(3),SUMHDZ(3),HDX(2),HDY(2),HDZ(2),WKAREA(850)
D      DATA MATA/729 * 0./

INITIAL

*    INPUT PARAMETER CONSTANTS
          A = 2.0
          P = 0.0
          W = 2*PI
          IDGT = 4
          G=0.0
          N=27
          M=1
          IA =27
          RUN = 2

*       INPUT JOINT LOCATIONS IN METERS
           JX0 = 0.0
           JY0 = 0.0
           JZ0 = 0.0
           JX1 = 0.0
           JY1 = 1.0
           JZ1 = 0.0
           JX2 = 0.0
           JY2 = 2.0
           JZ2 = 0.0

*       INPUT TORQUE CONSTANTS
           T0X = 0.0
           T0Y = 0.0
           T0Z = 0.0
           T1Y = 0.0
           T1Z = 0.0
           T2Y = 0.0
           T2Z = 0.0

*INPUT DISTANCE FROM CENTER OF LINK TO CENTER OF MASS FOR EACH LINK ENDS
           L(1,1) = 0.50
           L(1,2) = 0.50
           L(2,1) = 0.50
           L(2,2) = 0.50
```

```
            L(3,1) = 0.50
            L(3,2) = 0.50

*       INPUT MASS AT LINK ENDS IN KILOGRAMS
            MASS(1,1) = 2.5
            MASS(1,2) = 2.5
            MASS(2,1) = 2.5
            MASS(2,2) = 2.5
            MASS(3,1) = 2.5
            MASS(3,2) = 2.5

*       INPUT OMEGA AND OMEGA DOT
             DO 30 I = 1,3
                WX(I) = 0.0
                WY(I) = 0.0
                WZ(I) = 0.0
                WDX(I) = 0.0
                WDY(I) = 0.0
                WDZ(I) = 0.0
30          CONTINUE

*       INPUT INITIAL VALUES OF EULER ANGLE THETA AND CONVERT TO RADIANS
            ETHETX(1) = 90.0
             TX1 = ETHETX(1) * DEGRA
             ETHETY(1) = 0.0
             TY1 = ETHETY(1) * DEGRA
             ETHETZ(1) = 90.0
             TZ1 = ETHETZ(1) * DEGRA
             ETHETX(2) = 90.0
             TX2 = ETHETX(2) * DEGRA
             ETHETY(2) = 0.0
             TY2= ETHETY(2) * DEGRA
             ETHETZ(2) = 90.0
             TZ2 = ETHETZ(2) * DEGRA
             ETHETX(3) = 90.0
             TX3 = ETHETX(3) * DEGRA
             ETHETY(3) = 45.0
             TY3 = ETHETY(3) * DEGRA
             ETHETZ(3) = 45.0
             TZ3 = ETHETZ(3) * DEGRA

*       INPUT LOCATION OF LINK CENTERS OF GRAVITY
            LCOGX(1) = 0.0
            X1 = LCOGX(1)
            LCOGY(1) = 0.5
            Y1 = LCOGY(1)
            LCOGZ(1) = 0.0
            Z1 = LCOGZ(1)
            LCOGX(2) = 0.0
            X2 = LCOGX(2)
            LCOGY(2) = 1.5
            Y2 = LCOGY(2)
            LCOGZ(2) = 0.0
            Z2 = LCOGZ(2)
            LCOGX(3) = 0.0
            X3 = LCOGX(3)
            THERAD = ETHETY(3) * DEGRA
            LCOGY(3) = 2.0 + COS(THERAD) * L(3,1)
            Y3 = LCOGY(3)
            LCOGZ(3) = L(3,1) * SIN(THERAD)
            Z3 = LCOGZ(3)

*       INPUT MASS OF EACH LINK IN KG AND COMPUTE WEIGHTS IN NEWTONS
             MASS1 = 5.0
             MASS2 = 5.0
             MASS3 = 5.0
             W1 = MASS1*G
             W2 = MASS2*G
             W3 = MASS3*G
```

```
*     INPUT ACCELERATION OF JOINT ZERO
              AOX = 0.0
              AOY = 0.0
              AOZ = 0.0

DERIVATIVE

*       INPUT JOINT EQUATIONS
NOSORT

*       INITIALIZE MATRIX B TO ZERO
              DO 10 I = 1,27
                 MATB(I) = 0.0
10          CONTINUE

*       INPUT TORQUE AT JOINTS
                 T2X = -A*SIN (W*TIME +P)
                 T1X = A*SIN (W*TIME + P)

*           JOINT ZERO  AB = AG1 + (WD1 X RB/G1) + W1 X (W1 X RB/G1)
                 VECTA0(1) = WDX(1)
                 VECTA0(2) = WDY(1)
                 VECTA0(3) = WDZ(1)
                 RBG1(1) = JX0 - LCOGX(1)
                 RBG1(2) = JY0 - LCOGY(1)
                 RBG1(3) = JZ0 - LCOGZ(1)
              CALL CPROD(VECTA0,RBG1,MIA0,MJA0,MKA0)
                 VECTA0(1) = WX(1)
                 VECTA0(2) = WY(1)
                 VECTA0(3) = WZ(1)
              CALL CPROD(VECTA0,RBG1,MIB0,MJB0,MKB0)
                 VECTB0(1) = MIB0
                 VECTB0(2) = MJB0
                 VECTB0(3) = MKB0
              CALL CPROD(VECTA0,VECTB0,MIC0,MJC0,MKC0)

*    JOINT ONE EQUATIONS--- AA = AG1 + (WD1 X RA/G1) + W1 X (W1 X RA/G1)
                 VECTA1(1) = WDX(1)
                 VECTA1(2) = WDY(1)
                 VECTA1(3) = WDZ(1)
                 RAG1(1) = JX1 - LCOGX(1)
                 RAG1(2) = JY1 - LCOGY(1)
                 RAG1(3) = JZ1 - LCOGZ(1)
              CALL CPROD(VECTA1,RAG1,MIA1,MJA1,MKA1)
                 VECTA1(1) = WX(1)
                 VECTA1(2) = WY(1)
                 VECTA1(3) = WZ(1)
              CALL CPROD(VECTA1,RAG1,MIB1,MJB1,MKB1)
                 VECTB1(1) = MIB1
                 VECTB1(2) = MJB1
                 VECTB1(3) = MKB1
              CALL CPROD(VECTA1,VECTB1,MI[illegible],MJ[illegible],MK[illegible])

*      AB = AG2 + (WD[illegible] X RB[illegible]) + W[illegible] X (W[illegible] X RB[illegible])
                 VE[illegible]
                 VE[illegible]
                 VE[illegible]
                 RB[illegible]
                 RB[illegible]
                 RB[illegible]
              [illegible]
              [illegible]
```

```
                    CALL CPROD (VECTA1,VECTB1,MIC2,MJC2,MKC2)

* JOINT TWO EQUATIONS
*       AA = AG2 + (WD2 X RA/G2) + W2 X (W2 X RA/G2)
                       VECTA2(1) = WDX(2)
                       VECTA2(2) = WDY(2)
                       VECTA2(3) = WDZ(2)
                       RAG2(1) = JX2 - LCOGX(2)
                       RAG2(2) = JY2 - LCOGY(2)
                       RAG2(3) = JZ2 - LCOGZ(2)
                    CALL CPROD (VECTA2,RAG2,MIA3,MJA3,MKA3)
                       VECTA2(1) = WX(2)
                       VECTA2(2) = WY(2)
                       VECTA2(3) = WZ(2)
                    CALL CPROD (VECTA2,RAG2,MIB3,MJB3,MKB3)
                       VECTB2(1) = MIB3
                       VECTB2(2) = MJB3
                       VECTB2(3) = MKB3
                    CALL CPROD(VECTA2,VECTB2,MIC3,MJC3,MKC3)

*       AB = AG3 + (WD3 X RB/G3) + W3 X (W3 X RB/G3)
                       VECTA2(1) = WDX(3)
                       VECTA2(2) = WDY(3)
                       VECTA2(3) = WDZ(3)
                       RBG3(1) = JX2 - LCOGX(3)
                       RBG3(2) = JY2 - LCOGY(3)
                       RBG3(3) = JZ2 - LCOGZ(3)
                    CALL CPROD (VECTA2,RBG3,MIA4,MKA4,MKA4)
                       VECTA2(1) = WX(3)
                       VECTA2(2) = WY(3)
                       VECTA2(3) = WZ(3)
                    CALL CPROD (VECTA2,RBG3,MIB4,MJB4,MKB4)
                       VECTB2(1) = MIB4
                       VECTB2(2) = MJB4
                       VECTB2(3) = MKB4
                    CALL CPROD (VECTA2,VECTB2,MIC4,MJC4,MKC4)

*     SUM OF MOMENTS EQUATIONS
*         CONVERT EULER ANGLES FROM DEGREES TO RADIANS
            DO 40 I = 1,3
               THETXR(I) = ETHETX(I) * DEGRA
               THETYR(I) = ETHETY(I) * DEGRA
               THETZR(I) = ETHETZ(I) * DEGRA

*       COMPUTE HX H DOT X,HY,H DOT Y, HZ,H DOT Z
                       RX(I,1) = -L(I,1) * COS(THETXR(I))
                       RX(I,2) = L(I,2) * COS(THETXR(I))
                       RY(I,1) = -L(I,1) * COS(THETYR(I))
                       RY(I,2) = L(I,2) * COS(THETYR(I))
                       RZ(I,1) = -L(I,1) * COS(THETZR(I))
                       RZ(I,2) = L(I,2) * COS(THETZR(I))
               IXX(I,1)=MASS(I,1)*((RY(I,1)*RY(I,1))+(RZ(I,1)*RZ(I,1)))
               IXX(I,2)=MASS(I,2)*((RY(I,2)*RY(I,2))+(RZ(I,2)*RZ(I,2)))
               IXZ(I,1)=MASS(I,1) * RZ(I,1) * RX(I,1)
               IXZ(I,2)=MASS(I,2) * RZ(I,2) * RX(I,2)
               IXY(I,1)=MASS(I,1) * RX(I,1) * RY(I,1)
               IXY(I,2)=MASS(I,2) * RX(I,2) * RY(I,2)
                    HDX(1) = WDX(1)*IXX(I,1)-WDZ(I)*IXZ(I,1)-WDY(I)*IXY(I,1)
                    HDX(2) = WDX(2)*IXX(I,2)-WDZ(I)*IXZ(I,2)-WDY(I)*IXY(I,2)
               IYY(I,1) = MASS(I,1) * ((RX(I,1)*RX(I,1)) + (RZ(I,1)*RZ(I,1)))
               IYY(I,2) = MASS(I,2) * ((RX(I,2)*RX(I,2)) + (RZ(I,2)*RZ(I,2)))
               IYZ(I,1) = MASS(I,1) * RY(I,1) * RZ(I,1)
               IYZ(I,2) = MASS(I,2) * RY(I,2) * RZ(I,2)
                    HDY(1) = WDY(I)*IYY(I,1)-WDX(I)*IXY(I,1)-WDZ(I)*IYZ(I,1)
                    HDY(2) = WDY(I)*IYY(I,2)-WDX(I)*IXY(I,2)-WDZ(I)*IYZ(I,2)
               IZZ(I,1) = MASS(I,1) * ((RX(I,1)*RX(I,1)) + (RY(I,1)*RY(I,1)))
               IZZ(I,2) = MASS(I,2) * ((RX(I,2)*RX(I,2)) + (RY(I,2)*RY(I,2)))
                    HDZ(1) = WDZ(I)*IZZ(I,1)-WDX(I)*IXZ(I,1)-WDY(I)*IYZ(I,1)
                    HDZ(2) = WDZ(I)*IZZ(I,2)-WDX(I)*IXZ(I,2)-WDY(I)*IYZ(I,2)
```

```
          SUMHDX(I) = HDX(1) + HDX(2)
          SUMHDY(I) = HDY(1) + HDY(2)
          SUMHDZ(I) = HDZ(1) + HDZ(2)
40      CONTINUE

*    TEST TO SEE WHICH CONSTRAINT IS IN EFFECT 1,2 OR 3
          IF (RUN .EQ. 1) GO TO 1
          IF (RUN .EQ. 2) GO TO 2
          IF (RUN .EQ. 3) GO TO 3

*    INITIAIIZE MATRIX ACCORDING TO CONSTRAINT
1       DO 60 I = 1,18
              MATA(I,I) = 1.0
60      CONTINUE
          GO TO 4
2       DO 70 I = 1,9
              MATA(I,I) = 1.0
70      CONTINUE
          GO TO 7

*    ENTER CONSTANTS INTO MATRIX A
*     LINK ONE
*       SUM OF FORCES IN THE X DIRECTION
3        MATA(1,1) = 1.0
          MATA(1,4) = MASS1
          MATA(1,10) = -1.0

*       SUM OF FORCES IN Y DIRECTION
          MATA(2,2) = 1.0
          MATA(2,5) = MASS1
          MATA(2,11) = -1.0

*      SUM OF FORCES IN Z DIRECTION
          MATA(3,3) = 1.0
          MATA(3,6) = MASS1
          MATA(3,12) = -1.0

*       SUM OF FORCES LINK ONE EQUAL
          MATB(3) = -W1

*     EQUATIONS AT JOINT ZERO
*       IN THE X DIRECTION
          MATA(4,4) = 1.0
          MATA(4,8) = RBG1(3)
          MATA(4,9) = -RBG1(2)

          MATB(4)= AOX - MICO

*       IN THE Y DIRECTION
          MATA(5,5) = 1.0
          MATA(5,7) = -RBG1(3)
          MATA(5,9) = RBG1(1)

          MATB(5) = AOY - MJCO

*       IN THE Z DIRECTION
          MATA(6,6) = 1.0
          MATA(6,7) = RBG1(2)
          MATA(6,8) = -RBG1(1)

          MATB(6) = AOZ - MKCO

*    SUM OF MOMENTS EQUATIONS FOR LINK ONE IN THE X,Y,Z DIRECTIONS

          MATA(7,2) = RBG1(3)
          MATA(7,3) = -RBG1(2)
          MATA(7,7) = -(IXX(1,1) + IXX(1,2))
          MATA(7,8) = IXY(1,1) + IXY(1,2)
          MATA(7,9) = IXZ(1,1) + IXZ(1,2)
```

```
            MATA(7,11) = -RAG1(3)
            MATA(7,12) = RAG1(2)

            MATB(7) = T1X - T0X

            MATA(8,1) = -RBG1(3)
            MATA(8,3)= RBG1(1)
            MATA(8,7) = IXY(1,1) + IXY(1,2)
            MATA(8,8) = -(IYY(1,1) + IYY(1,2))
            MATA(8,9) = IYZ(1,1) + IYZ(1,2)
            MATA(8,10) = RAG1(3)
            MATA(8,12) = -RAG1(1)

            MATB(8) = T1Y - T0Y

            MATA(9,1) = RBG1(2)
            MATA(9,2) = -RBG1(1)
            MATA(9,7) = IXZ(1,1) + IXZ(1,2)
            MATA(9,8) = IYZ(1,1) + IYZ(1,2)
            MATA(9,9) = -(IZZ(1,1) + IZZ(1,2))
            MATA(9,10) = -RAG1(2)
            MATA(9,11) = RAG1(1)

            MATB(9) = T1Z - T0Z

*   LINK TWO
*      SUM OF FORCES IN X DIRECTION
7        MATA(10,10) = 1.0
            MATA(10,13) = MASS2
            MATA(10,19) = -1.0

*      SUM OF FORCES IN THE Y DIRECTION
            MATA(11,11) = 1.0
            MATA(11,14) = MASS2
            MATA(11,20) = -1.0

*      SUM OF FORCES IN THE Z DIRECTION
            MATA(12,12) = 1.0
            MATA(12,15) = MASS2
            MATA(12,21) = -1.0

*      SUM OF FORCES LINK TWO EQUAL
          MATB(12) = -W2

*     EQUATIONS AT JOINT ONE
*       IN THE X DIRECTION
            MATA(13,4) = -1.0
            MATA(13,8) = -RAG1(3)
            MATA(13,9) = RAG1(2)
            MATA(13,13) = 1.0
            MATA(13,17) = RBG2(3)
            MATA(13,18) = -RBG2(2)

            MATB(13) = MIC1 - MIC2

*       IN THE Y DIRECTION
            MATA(14,5) = -1.0
            MATA(14,7) = RAG1(3)
            MATA(14,9) = -RAG1(1)
            MATA(14,14) = 1.0
            MATA(14,16) = -RBG2(3)
            MATA(14,18) = RBG2(1)

            MATB(14) = MJC1 - MJC2

*       IN THE Z DIRECTION
            MATA(15,6) = -1.0
            MATA(15,7) = -RAG1(2)
            MATA(15,8) = RAG1(1)
```

```
          MATA(15,15) = 1.0
          MATA(15,16) = RBG2(2)
          MATA(15,17) = -RBG2(1)

          MATB(15) = MKC1 - MKC2

*   SUM OF MOMENTS EQUATIONS FOR LINK TWO IN THE X,Y,Z DIRECTIONS
          MATA(16,11) = RBG2(3)
          MATA(16,12) = -RBG2(2)
          MATA(16,16) = -(IXX(2,1) + IXX(2,2))
          MATA(16,17) = IXY(2,1) + IXY(2,2)
          MATA(16,18) = IXZ(2,1) + IXZ(2,2)
          MATA(16,20) = -RAG2(3)
          MATA(16,21) = RAG2(2)
          MATB(16) = -T1X + T2X
          IF(RUN .EQ. 2) GO TO 11

          MATA(17,10) = -RBG2(3)
          MATA(17,12) = RBG2(1)
          MATA(17,16) = IXY(2,1) + IXY(2,2)
          MATA(17,17) = -(IYY(2,1) + IYY(2,2))
          MATA(17,18) = IYZ(2,1) + IYZ(2,2)
          MATA(17,19) = RAG2(3)
          MATA(17,21) = -RAG2(1)

          MATB(17) =   - T1Y + T2Y

          MATA(18,10) = RBG2(2)
          MATA(18,11) = -RBG2(1)
          MATA(18,16) = IXZ(2,1) + IXZ(2,2)
          MATA(18,17) = IYZ(2,1) + IYZ(2,2)
          MATA(18,18) = -(IZZ(2,1) + IZZ(2,2))
          MATA(18,19) = -RAG2(2)
          MATA(18,20) = RAG2(1)

          MATB(18) =   - T1Z + T2Z

          IF (RUN .EQ. 3) GO TO 4
11      MATA(17,17) = 1.0
          MATA(18,18) = 1.0

*   LINK THREE
*      SUM OF FORCES IN THE X DIRECTION
4        MATA(19,19) = 1.0
          MATA(19,22) = MASS3

*      SUM OF FORCES IN THE Y DIRECTION
          MATA(20,20) = 1.0
          MATA(20,23) = MASS3

*      SUM OF FORCES IN THE Z DIRECTION
          MATA(21,21) = 1.0
          MATA(21,24) = MASS3

        MATB(21) = -W3
*   EQUATIONS AT JOINT TWO
*      IN THE X DIRECTION
          MATA(22,13) = -1.0
          MATA(22,17) = -RAG2(3)
          MATA(22,18) = RAG2(2)
          MATA(22,22) = 1.0
          MATA(22,26) = RBG3(3)
          MATA(22,27) = -RBG3(2)

          MATB(22) = MIC3 - MIC4

*      IN THE Y DIRECTION
          MATA(23,14) = -1.0
          MATA(23,16) = RAG2(3)
```

```
            MATA(23,18) = -RAG2(1)
            MATA(23,23) = 1.0
            MATA(23,25) = -RBG3(3)
            MATA(23,27) = RBG3(1)

            MATB(23) = MJC3 - MJC4

*       IN THE Z DIRECTION
            MATA(24,15) = -1.0
            MATA(24,16) = -RAG2(2)
            MATA(24,17) = RAG2(1)
            MATA(24,24) = 1.0
            MATA(24,25) = RBG3(2)
            MATA(24,26) = -RBG3(1)

            MATB(24) = MKC3 - MKC4

*     SUM OF MOMENTS EQUATIONS FOR LINK THREE IN THE X,Y,Z DIRECTIONS
            MATA(25,20) = RBG3(3)
            MATA(25,21) = -RBG3(2)
            MATA(25,25) = -(IXX(3,1) + IXX(3,2))
            MATA(25,26) = IXY(3,1) + IXY(3,2)
            MATA(25,27) = IXZ(3,1) + IXZ(3,2)
            MATB(25) = - T2X
            IF(RUN .EQ. 1 .OR. RUN .EQ. 2) GO TO 12

            MATA(26,19) = -RBG3(3)
            MATA(26,21) = RBG3(1)
            MATA(26,25) = IXY(3,1) + IXY(3,2)
            MATA(26,26) = -(IYY(3,1) + IYY(3,2))
            MATA(26,27) = IYZ(3,1) + IYZ(3,2)

            MATB(26) = - T2Y

            MATA(27,19) = RBG3(2)
            MATA(27,20) = -RBG3(1)
            MATA(27,25) = IXZ(3,1) + IXZ(3,2)
            MATA(27,26) = IYZ(3,1) + IYZ(3,2)
            MATA(27,27) = -(IZZ(3,1) + IZZ(3,2))

           MATB(27) = - T2Z

            IF (RUN .EQ. 3) GO TO 13
12        MATA(26,26) = 1.0
            MATA(27,27) = 1.0

* CALL EQUATION SOLVER PROGRAM FROM IMSL
13     CALL LEQT2F(MATA,M,N,IA,MATB,IDGT,WKAREA,IER)
         IF (IER .NE. 0) CALL ENDJOB

* FIND LCOGX,LCOGY,LCOGZ,THETA VALUES,WX,WY,WZ
         IF(RUN .EQ. 1) GO TO 6
         IF (RUN .EQ. 2) GO TO 9

*      LINK ONE
            AX1 = MATB(4)
            VELX1 = INTGRL(0,AX1)
            LCOGX1 = INTGRL(X1,VELX1)
            LCOGX(1) = LCOGX1
            AY1 = MATB(5)
            VELY1 = INTGRL( ,AY1)
            LCOGY1 = INTGRL(Y1,VELY1)
            LCOGY(1) = LCOGY1
            AZ1 = MATB(6)
            VELZ1 = INTGRL( ,AZ1)
            LCOGZ1 = INTGRL(Z1,VELZ1)
            LCOGZ(1) = LCOGZ1
            WD1X = MATB(7)
```

```
          W1X = INTGRL(0,WD1X)
          THEXR1 = INTGRL(TY1,W1X)
          JX0= LCOGX(1) - L(1,1) * COS(TX1)
          WDX(1) = WD1X
          WX(1) = W1X
          CTHETX(1) = THEXR1 * RADEG
          ETHETY(1) = CTHETX(1)
          WD1Y = MATB(8)
          W1Y = INTGRL(0,WD1Y)
          THEYR1 = INTGRL(0.,W1Y)
          JY0 = LCOGY(1) - L(1,1) * COS(THEXR1)
          WDY(1) = WD1Y
          WY(1) = W1Y
          CTHETY(1) = THEYR1 * RADEG
          WD1Z = MATB(9)
          W1Z = INTGRL(0,WD1Z)
          THEZR1 = INTGRL(0.,W1Z)
          WDZ(1) = WD1Z
          WZ(1) = W1Z
          CTHETZ(1) = THEZR1 * RADEG
          ETHETZ(1) = 90.0 - CTHETX(1)
          ETHEZ1 = ETHETZ(1) * DEGRA
          JZ0 = LCOGZ(1) - L(1,1) * COS(ETHEZ1)

*     LINK TWO
9       AX2 = MATB(13)
          VELX2 = INTGRL(0.,AX2)
          LCOGX2 = INTGRL(X2,VELX2)
          LCOGX(2) = LCOGX2
          AY2 = MATB(14)
          VELY2 = INTGRL(0.,AY2)
          LCOGY2 = INTGRL(Y2,VELY2)
          LCOGY(2) = LCOGY2
          AZ2 = MATB(15)
          VELZ2 = INTGRL(0.,AZ2)
          LCOGZ2 = INTGRL(Z2,VELZ2)
          LCOGZ(2) = LCOGZ2
          WD2X = MATB(16)
          W2X = INTGRL(0.,WD2X)
          THEXR2 = INTGRL(TY2,W2X)
          JX1 = LCOGX(2) - L(2,1) * COS(TX2)
          WDX(2) = WD2X
          WX(2) = W2X
          CTHETX(2) = THEXR2 * RADEG
          ETHETY(2) = CTHETX(2)
          WD2Y = MATB(17)
          W2Y = INTGRL(0.,WD2Y)
          THEYR2 = INTGRL(0.,W2Y)
          JY1 = LCOGY(2) - L(2,1) * COS(THEXR2)
          WDY(2) = WD2Y
          WY(2) = W2Y
          CTHETY(2) =THEYR2 * RADEG
          WD2Z = MATB(18)
          W2Z = INTGRL(0.,WD2Z)
          THEZR2 = INTGRL(0.,W2Z)
          WDZ(2) = WD2Z
          WZ(2) = W2Z
          CTHETZ(2) = THEZR2 * RADEG
          ETHETZ(2) = 90.0 - CTHETX(2)
          ETHEZ2 = ETHETZ(2) * DEGRA
          JZ1 = LCOGZ(2) - L(2,1) * COS(ETHEZ2)

*     LINK THREE
9       AX3 = MATB(22)
          VELX3 = INTGRL(0.,AX3)
          LCOGX3 = INTGRL(X3,VELX3)
          LCOGX(3) = LCOGX3
          AY3 = MATB(23)
          VELY3 = INTGRL(0.,AY3)
```

```
          LCOGY3 = INTGRL(Y3,VELY3)
          LCOGY(3) = LCOGY3
          AZ3 = MATB(24)
          VELZ3 = INTGRL(0.,AZ3)
          LCOGZ3 = INTGRL(Z3,VELZ3)
          LCOGZ(3) = LCOGZ3
          WD3X = MATB(25)
          W3X = INTGRL(0.,WD3X)
          THEXR3 = INTGRL(TY3,W3X)
          JX2 = LCOGX(3) - L(3,1) * COS(TX3)
          WDX(3) = WD3X
          WX(3) = W3X
          CTHETX(3) = THEXR3 * RADEG
          ETHETY(3) = CTHETX(3)
          WD3Y = MATB(26)
          W3Y = INTGRL(0.,WD3Y)
          THEYR3 = INTGRL(0.,W3Y)
          JY2 = LCOGY(3) - L(3,1) * COS(THEXR3)
          WDY(3) = WD3Y
          WY(3) = W3Y
          CTHETY(3) = THEYR3 * RADEG
          WD3Z = MATB(27)
          W3Z = INTGRL(0.,WD3Z)
          THEZR3 = INTGRL(0.,W3Z)
          WDZ(3) = WD3Z
          WZ(3) = W3Z
          CTHETZ(3) = THEZR3 * RADEG
          ETHETZ(3) = 90.0 - CTHETX(3)
          ETHEZ3 = ETHETZ(3) * DEGRA
          JZ2 = LCOGZ(3) - L(3,1) * COS(ETHEZ3)
DYNAMIC

*  COMPUTE THEORITICAL TORQUE,T1X AND T2X
          Y = L(3,1) * COS(THEXR3)
          Z = L(3,1) * SIN(THEXR3)
          FZ2 =-MASS3*AZ3
          FY2 = -MASS3 * AY3
          FZ1 = FZ2 - MASS2 * AZ2
          FY1 = FY2 - MASS2 * AY2
          TORY2X = (MASS3 * L(3,2)**2)*WDX(3) -(FZ2 * Y)+ (FY2 * Z)
          TORY1X = (MASS2*L(2,1)**2)*WDX(2)+TORY2X-FZ1*COS(THEXR2)...
          *L(2,1)+FY1*SIN(THEXR2)*L(2,1)-FZ2*L(2,2)*COS(THEXR2)+FY2...
          *SIN(THEXR2)*L(2,2)
*     COMPUTE ERROR BETWEEN COMPUTED AND INPUTEDVALUES OF TORQUE AT
*    JOINT ONE AND TWO
          ERRT2X = ((TORY2X-T2X)/ 4.7553) * 100.
          ERRT1X = ((TORY1X-T1X)/ 4.7553) * 100.
END
STOP
FORTRAN

*     SUBROUTINE TO COMPUTE THE CROSS PRODUCT OF TWO VECTORS

          SUBROUTINE CPROD(VECTA,VECTB,MI,MJ,MK)
               IMPLICIT REAL*8 (A-Z)
               DIMENSION VECTA(3),VECTB(3)
               MI = VECTA(2) * VECTB(3) - VECTA(3) * VECTB(2)
               MJ = VECTA(3) * VECTB(1) - VECTA(1) * VECTB(3)
               MK = VECTA(1) * VECTB(2) - VECTA(2) * VECTB(1)
           RETURN
           END
```

LIST OF REFERENCES

1. Craig, J. J., Introduction to Robotic Mechanisms and Control, pp.146-149, Addison-Wesley Co.,1986.

2. Walker, M. W., and Orin, D.E.,"Efficient Dynamic Computer Simulation of Robotic Mechanisms," ASME J. of Dynamic Systems, Measurements and Control, v. 104, pp. 205-211.

3. Luh, J. Y. S., Walker, M. W., and Paul, R. P. C., "On-line Computational Scheme for Mechanical Manipulators," ASME J. of Dynamic Systems, Measurements and Control, v. 102, pp. 69-76, June 1980.

4. Hollerbach, J. M., "A Recursive Lagrangian Formulation of Manipulative Dynamics and a Comparative Study of Dynamics Formulation Complexity," IEEE Trans SYST Man, Cybern, v. SMC10 no. 11, pp. 730-736, 1980.

5. Paul, R., "The Mathematics of Computer Controlled Manipulators," Proceedings Joint Control Conference, v. I, pp. 124-131, 1977.

6. Williams, R. J., and Seireg, A., "Interactive Modeling and Analysis of Open and Closed Loop Dynamic Systems With Redundant Activators," ASME J. of Mechanical Design, v. 101, no. 3, pp.407-416, July 1979.

7. Kane, T., and Levinson, D. A., "The Use of Kane's Dynamical Equations in Robotics," International J. of Robotics Research, v. 2, no. 3, pp. 3-21, Fall 1983.

8. McCarthy, W. F., Simulation of High Speed Motion of Rigid, Revolute Mechanisms, M. S. Thesis, Naval Postgraduate School, Monterey, California, December 1985.

9. Klein, C. A., and Patterson, M. R., "Computer Coordination of Limb Motion for Locomotion of a Multiple-Armed Robot for Space Assembly," IEEE Trans Syst Man, Cybern., v. SMC-12, pp. 913-919, November/December 1982.

10. Paul, R. P., and Stevenson, C. N., "Kinematics of Robot Wrists," International J. of Robotics Res., v. 2, no. 1, pp. 31-38, Spring 1983.

11. Whitney, D. E., "Resolved Motion Rate Control of Manipulators and Human Prothesis," IEEE Trans. Man-Mach. Syst., v. 10, no. 2, pp. 47-53, June 1969.

12. Nakamura, Y., and Hanafusa, H., "Inverse Kinematic Solutions with Singularity Robustness for Robot Manipulator Control," ASME-Robotics and Manufacturing Automation Proceedings, pp. 193-204, 1983.

13. Barker, K. L., "Kinematic Equations for Resolved Rate Control of an Industrial Robot Arm," NASA Tech. Memo., 85685, pp. 1-35, November 1983.

14. Barker, K. L., Houck, J. A., and Carzoo, S. W., "Translational Control of a Graphically Simulated Robot Arm by Kinematic Rate Equatiions that Overcome Elbow Joint Singularity," NASA Tech. Paper, 2376, December 1984.

15. Waldron, K. J., Wang, S., and Bolin, S. J., "A Study of the Jacobian Matrix of Serial Manipulators," ASME Design Engineering Technical Conference, Cambridge, Massachusetts, October 7-10, 1984.

16. Yoshikawa,T., "Analysis and Control of Robot Manipulators with Redundancy," in Robotics Research-First International Symposium, ed. M. Brady and R. Paul, Cambridge,Massachusetts, pp. 735-748, 1984.

17. Klein, C. A., and Huang, C. H., "Review of Pseudo-inverse control for use with Kinematically Redundant Manipulators," IECC Trans Syst. Man., Cybern., v. SMC-13, no. 3, pp. 245-250, March-April 1983.

18. Uchiyama, M., "Study on Dynamic Control of Artificial Arm -part 1," Trans. of JSME, v. 45(c), pp. 314-322, 1979.

19. Hollerbach, J. M., "Optimum Kinematic Design for a Seven Degree of Freedom Manipulator," 2nd International Symposium of Robotics Research, Kyoto, Japan, August 20-23, pp. 349-356, 1984.

20. Hollerbach, J. M., and Suh, K. C., "Redundancy Resolution of Manipulators Through Torque Optimization," IEEE International Conference on Robotics and Automation, pp. 1016-1021, March 25-28 1985.

21. Luh, J. Y. S., and Gu, Y. L., "Industrial Robots with Seven Joints," IEEE International Conference on Robotics and Automation, pp. 1010-1015, March 25-28, 1985.

22. Yoshikawa, T., "Manipulability of Robotic Mechanisms," 2nd International Symposium of Robotics Research, Kyoto, Japan, pp. 91-98, August 20-23, 1984.

23. Shih, L. Y., "Redundant Anthropomorphic Manipulator Subject to Kinematic Constraint," IEEE Proceedings of the International Conference on Systems,Man, Cybern, v. 1, pp. 676-679, 1983.

24. Takase, K., Inoue, H., and Sato, K., "The Design of an Articulated Manipulator with Torque Control Stability," Proceedings of the 4th International Symposium on Industrial Robots, pp. 261-270, November 19-21, 1974.

25. Trevelyan, J. P., Kouesi, P. P., Ong, M., and Elford, D., "ET: A Wrist Mechanism without Singular Positions," International Journal of Robotics Research, v. 4, no. 4, pp. 71-85, Winter 1986.

26. Meriam, J. L., Engineering Mechanics,v. 2, Dynamics, (English version), John Wiley and Sons, 1978.

INITIAL DISTRIBUTION LIST

		NO. COPIES
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Department Chairman, Code 69 Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
4.	Professor D. L. Smith, Code 69Sm Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	5
5.	Professor Nunn, Code 69Nn Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
6.	Professor Chang, Code 69Ck Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
7.	Ms. Mary Lacey Naval Surface Weapons Center White Oack, Code R402 Robotics Research and Development Center 10908 New Hampshire Avenue Silver Springs, Maryland 20903-5000	2
8	Lieutenant K. Mohammed, USN Supervisor of Shipbuilding Conversion and Repair, USN Flushing and Washington Avenues Brooklyn, New York 11251	2

END

3-87

DTIC